PriPrune: Quantifying and Preserving Privacy in Pruned Federated Learning

TIANYUE CHU*, IMDEA Networks Institute and Universidad Carlos III de Madrid, Spain
MENGWEI YANG*, University of California, Irvine, USA
NIKOLAOS LAOUTARIS, IMDEA Networks Institute, Spain
ATHINA MARKOPOULOU, University of California, Irvine, USA

Model pruning has been proposed as a technique for reducing the size and complexity of Federated learning (FL) models. By making local models coarser, pruning is intuitively expected to improve protection against privacy attacks. However, the level of this expected privacy protection has not been previously characterized, or optimized jointly with utility. In this paper, we first characterize the privacy offered by pruning. We establish information-theoretic upper bounds on the information leakage from pruned FL and we experimentally validate them under state-of-the-art privacy attacks across different FL pruning schemes. Second, we introduce *PriPrune*– a privacy-aware algorithm for pruning in FL. *PriPrune* uses defense pruning masks, which can be applied locally after *any* pruning algorithm, and adapts the defense pruning rate to jointly optimize privacy and accuracy. Another key idea in the design of *PriPrune* is *pseudo-pruning:* it undergoes defense pruning within the local model and only sends the pruned model to the server; while the weights pruned out by defense mask are withheld locally for future local training rather than being removed. We show that *PriPrune* significantly improves the privacy-accuracy tradeoff compared to state-of-the-art pruned FL schemes. For example, on the FEMNIST dataset, *PriPrune* improves the privacy of *PruneFL* by 45.5% without reducing accuracy.

CCS Concepts: • **Do Not Use This Code → Generate the Correct Terms for Your Paper**; *Generate the Correct Terms for Your Paper*; Generate the Correct Terms for Your Paper; Generate the Correct Terms for Your Paper.

Additional Key Words and Phrases: Federated learning, Privacy, Model pruning

ACM Reference Format:
Tianyue Chu, Mengwei Yang, Nikolaos Laoutaris, and Athina Markopoulou. 2018. PriPrune: Quantifying and Preserving Privacy in Pruned Federated Learning. *J. ACM* 37, 4, Article 111 (August 2018), 32 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Federated Learning (FL) has emerged as the predominant paradigm for distributed machine learning across a multitude of user devices [18, 26]. It is known to have several benefits in terms of reducing the communication, computation and storage costs for the users, training better global models, and raising the bar for privacy by not sharing the local data. FL allows users to train models locally on their (user) devices without revealing their local data but instead collaborate by sharing only model updates that can be combined to build a global model through a central server. A typical FL scenario

*Both authors contributed equally to the paper.

Authors' addresses: Tianyue Chu, IMDEA Networks Institute and Universidad Carlos III de Madrid, Madrid, Spain, tianyue.chu@imdea.org; Mengwei Yang, University of California, Irvine, Irvine, USA, mengwey@uci.edu; Nikolaos Laoutaris, IMDEA Networks Institute, Madrid, Spain, nikolaos.laoutaris@imdea.org; Athina Markopoulou, University of California, Irvine, Irvine, USA, athina@uci.edu.

Manuscript submitted to ACM

involves numerous users and resource-constrained devices [23] making it challenging to perform resource-intensive tasks like training Deep Neural Networks (DNNs) on them.

In parallel, significant efforts have been put toward optimizing sparse DNNs to create lightweight models suitable for training on edge devices [6, 14, 24]. Model pruning, which involves the removal of a certain percentage of parameters, has gained significant attention [13, 20, 28]. Its primary objective is to derive a sparse DNNs model that decreases the computational requirements and enhances efficiency without compromising accuracy [10]. Several model pruning techniques have been developed for FL to optimize model sparsity under given constraints on the processing and communication capabilities of devices, especially under heterogeneous clients, [3, 17].

It is intuitively expected that by reducing the number of weights of FL models, the pruning method should, as a side-effect, also improve their resistance to reconstruction attacks. These attacks aim at using these weights to reverse-engineer the model and obtain information about the input data; a.k.a. gradient inversion attacks launched by the server. Prior work on model pruning focuses solely on the scalability/accuracy trade-off and thus neither considers nor quantifies the privacy impact of pruning on FL. To the best of our knowledge, our paper is the first to consider privacy in the context of FL model pruning, and to address the following key questions: **Q1.** *Can we quantify privacy in FL model pruning?* **Q2.** *How can we further optimize pruning for privacy, and jointly with accuracy?* To address these questions, we make the following two contributions.

First, **we theoretically and empirically quantify privacy leakage in model pruning.** We derive information-theoretic upper bounds on the amount of information revealed about any single user's dataset via model updates, in *any* pruned FL scheme. This is the first theoretical characterization of privacy leakage in pruned FL models. We also conduct a comprehensive empirical evaluation that quantifies the actual amount of privacy leakage of six different pruning schemes, considering several state-of-the-art privacy attacks. Within the family of gradient inversion attacks (including Deep Leakage from Gradients [45] and Gradient Inversion (GI) [11]), we also design a novel privacy attack (referred to as Sparse Gradient Inversion, or SGI), specifically tailored to exploit vulnerabilities inherent in FL model pruning. This evaluation combined with our theoretical analysis provides valuable insights into the choices and parameters that affect the privacy provided by pruning.

Building upon these insights, we make our second contribution: we design *PriPrune*– **a privacy-preserving pruning mechanism.** *PriPrune* defends against gradient inversion attacks in pruned FL, by performing additional local (defense) pruning, after any (base) FL pruning scheme. *PriPrune* applies a personalized defense mask and adapts the defense pruning rate, so as to jointly optimize model accuracy *and* privacy, using back-propagation augmented with Gumbel Softmax Sampling. Another key idea in the design of *PriPrune* is what we refer to as *Pseudo-Pruning*: it entails pruning with the defense mask within the local model and transmitting the pruned model to the server, thereby enhancing privacy. However, the weights pruned out by the defense mask in the local model are not discarded; instead, they are retained locally for subsequent rounds of local training, thereby preserving model accuracy. We evaluate *PriPrune* via comprehensive experiments across various FL pruning schemes, a state-of-the-art attack and several benchmark datasets. We show that it achieves a significantly better privacy-accuracy tradeoff than privacy-unaware pruned FL baselines. For example, on the FEMNIST dataset, *PriPrune* improves the privacy *PruneFL* [17] by 45.5% without imposing any accuracy loss.

2 RELATED WORK

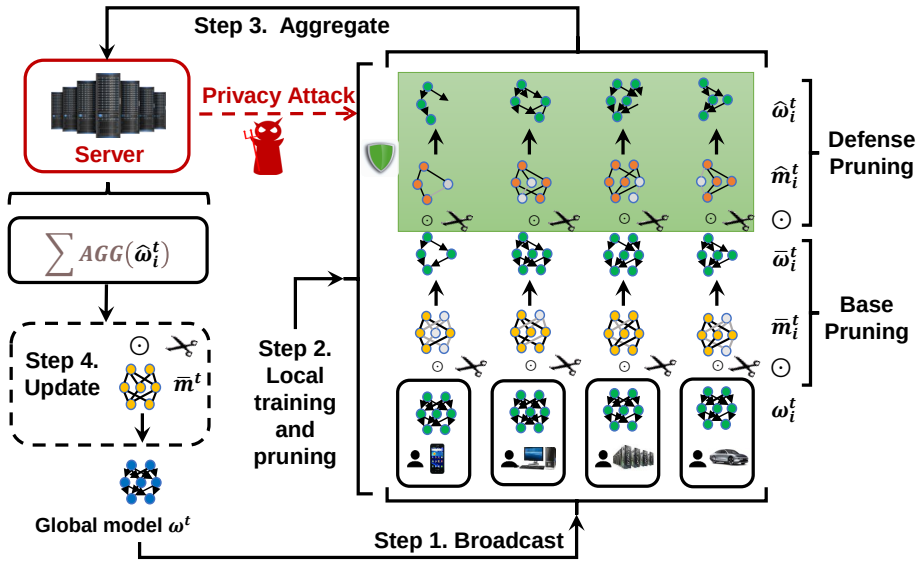


Fig. 1. FL with model pruning, under a privacy attack by an honest-but-curious server. Step 1: the server broadcasts the global model with weight w^t to users. Step 2: After local training, user i employs base pruning with pruning mask $\bar{m}_i^t$, resulting in a pruned model with weights $\bar{w}_i^t$. The user then applies a defense pruning technique, yielding a pruned model with weights $\hat{w}_i^t$. Step 3, the server aggregates the model updates and launches a gradient inversion attack upon receiving $\hat{w}_i^t$. Step 4: the server applies global pruning with pruning mask $\bar{m}^t$, and updates the global model accordingly. Notice *Base* pruning can be *any* FL pruning scheme. *Defense* pruning is introduced, in this paper, specifically to protect against privacy attacks.

Model Pruning in FL. State-of-the-art research for model pruning in FL utilizes two main approaches: one involves the pruning mask being decided on the server side and the client only using this mask without changing it, and the other involves the collaborative selection of the pruning mask by both the user and the server side. *PruneFL* [17] follows the first approach., aiming to reduce the size and complexity of FL models without compromising accuracy. It achieves this by removing less important weights and connections in each layer of the neural network based on their magnitude and importance scores. However, the pruning mask is only decided by the server without considering user-side local representations. *LotteryFL* [22] belongs to the second approach, introducing a method that allows users to maintain local representations by selecting a subset of the global network using personalized masks. However, the system of sparse models it produces performs well only on local datasets. *FedDST* [3] involves dynamic sparse training on both the client and server sides, extracting and training sparse sub-networks. Other recent state-of-the-art studies have introduced various gradient-based one-shot pruning algorithms at initialization. *Snip* [21] leverages connection sensitivity to retain critical connections and remove less significant ones, aiming to limit the loss due to pruning. On the other hand, *GraSP* [38] focuses on preserving the gradient flow through the network by scoring weights based on the Hessian-gradient product. *SynFlow* [34], being a data-agnostic pruning algorithm, emphasizes preserving the total flow of synaptic strengths through the neural network to achieve Maximal Critical Compression during initialization. More importantly, none of them considered privacy implications in their design, as *PriPrune* does.

Reconstruction Attacks in FL. In terms of privacy leakage, communicating gradients throughout the training process in federated learning can reveal sensitive information about the participants. *Deep leakage from gradients* (DLG)

[45] demonstrated that sharing the gradients can leak private training data, including images and text. *Improved DLG* (iDLG) [44], presented an analytical procedure to extract the ground-truth labels from the shared gradients and showed the advantages of iDLG over DLG. *Inverting Gradients* [11] introduced cosine similarity as a cost function in their reconstruction attacks and employed total variation loss $\mathcal{L}_{\mathcal{TV}}$ as an image prior. *Inverting Gradients* [11] and *GradInversion* [41] demonstrated the capability to reconstruct high-resolution images with increased batch sizes. The reconstruction quality of the attack was improved in *Reconstruction From Obfuscated Gradient* (ROG) [42]. ROG [42] proposed conducting the reconstruction optimization in low dimensional space and trained a neural network as a postprocessing module.

While these privacy attacks have demonstrated the privacy vulnerabilities of standard federated learning, none of them has considered pruned federated learning. Therefore, we do not currently know how effective existing attacks are against the sparser models from pruned FL.

Privacy-Preserving Techniques in FL. FL is particularly vulnerable to the aforementioned inverting-the-gradient type of reconstruction attacks, since it relies on gradient updates to operate. There exists a large body of literature on privacy-enhancing methods that protect against this type of attacks in FL. The adversary is typically an "honest-but-curious" server, *i.e.,* a server that participates correctly in the FL protocol, but tries to infer training data of individual clients based on their gradient updates. One family of defense approaches is based on differential privacy (DP) [7, 8, 12, 36, 40, 43], which can be applied locally (at the clients), centrally (at the server), or in a distributed way. In the local DP model for FL, each client adds noise to the local update before sharing with the server, which makes it difficult for the server to invert the true gradient and infer the client's training data. However, local DP is known to significantly degrade utility, in this case model accuracy. Another defense approach applied to FL is Secure Aggregation (SecAgg) [4, 32, 33, 37, 39]. SecAgg ensures that individual updates are encrypted (thus preventing the server from inverting the gradients of individual users), while their aggregate can still be decrypted (thus enabling the server to train a good model). SecAgg is powerful but has computational overhead, although significant advances have been done recently.

Both DP and SecAgg are important but orthogonal to the underlying FL and can be combined with a number of FL protocols, including all the pruning protocols we consider as base in this study. Our proposed *PriPrune* can also be combined with DP and/or SA. For instance, after applying *PriPrune* to base pruning, clients can add local DP to the pruned updates before sending them to the server, and/or they can encrypt their pruned updates to enable SA. This paper focuses on pruning itself and the privacy-utility tradeoff that can be achieved, *before* adding orthogonal defense techniques. Future work could explore adding DP and/or SecAgg, which are out of the scope of this paper.

Recent works like *FedMap* [15] and *Fed-LTP* [30] discuss privacy specifically in the context of pruned FL. FedMap, which appeared after this submission, enhances privacy by training models from scratch and by avoiding the sharing of detailed pruning masks. However, it does not quantify the privacy leakage that may still occur from the pruned models. *FedMap* can be considered as a base pruning technique, and *PriPrune* can still be applied to protect it. *Fed-LTP*, on the other hand, combines Lottery Ticket Pruning with Zero-concentrated Differential Privacy (zCDP) to enhance privacy. Integrating zCDP, or any type of DP, with our proposed *PriPrune* is orthogonal to our core contribution, as discussed above, and can be explored as part of future research.

3 PROBLEM SETUP

Federated Learning (FL) with Pruning. We consider a general FL system with multiple users and one server, employing *any* model pruning scheme on the users and/or the server. We refer to pruning at this stage as *base* pruning

Algorithm 1 FL with Pruning under Privacy Attack

Given: T number of global training rounds; N number of users indexed by i; E number of local epochs; the server aims to reconstruct the local data of the target user.
Server executes:
Initialize $\boldsymbol{w}^0$
for $t \leftarrow 1$ **to** T **do**
  for each user $i \in N$ **in parallel do**
    $\hat{\boldsymbol{w}}_i^t \leftarrow \mathit{UserUpdate}(\boldsymbol{w}^{t-1}, \boldsymbol{m}^{t-1}, i)$
    if user i is targeted **then**
      $\nabla \hat{\boldsymbol{w}}_i^t \leftarrow \hat{\boldsymbol{w}}_i^t - \hat{\boldsymbol{w}}_i^{t-1}$
      Privacy Attack on $(F(\mathcal{D}_i; \hat{\boldsymbol{w}}_i^t),\ \hat{\boldsymbol{w}}_i^t,\ \nabla \hat{\boldsymbol{w}}_i^t)$
    end if
  end for
  $\boldsymbol{w}^t \leftarrow \left[\sum_{i=1}^{I} \frac{\|\mathcal{D}_i\|}{\|\mathcal{D}\|} \hat{\boldsymbol{w}}_i^t\right] \odot \bar{\boldsymbol{m}}^t$;
end for
function $\mathit{UserUpdate}(\boldsymbol{w}^{t-1}, \boldsymbol{m}^{t-1}, i)$:
for local epoch $e \leftarrow 1$ **to** E **do**
  Local Training and Base Pruning: $\bar{\boldsymbol{w}}_i^t \leftarrow$ Eq. (1)
  Defense Pruning: $\hat{\boldsymbol{w}}_i^t \leftarrow \bar{\boldsymbol{w}}_i^t \odot \hat{\boldsymbol{m}}_i^t$
end for
Return $\hat{\boldsymbol{w}}_i^t$ to server

with base pruning rate $\bar{p}$, and it can be any state-of-the-art pruning FL scheme [3, 17]. This is depicted in Fig. 1 and formalized in Algorithm 1.

The goal is to train a global model $F(\mathcal{D}, \boldsymbol{w})$ through FL. At the beginning of round t, user i with the local data $\mathcal{D}_i$ and labels y_i has local model weights $\boldsymbol{w}_i^t$; the server broadcasts the most recent global model $F(\mathcal{D}, \boldsymbol{w}^t)$ to all participating users. Then the round proceeds as follows:

Local Training and Base Pruning: Each user i utilizes base pruning strategy $\mathsf{P}_i(\cdot)$ with base pruning masks $\bar{\boldsymbol{m}}_i^t$ on its trained local model $F(\mathcal{D}_i, \boldsymbol{w}_i^t)$:

$$\bar{\boldsymbol{w}}_i^t := \boldsymbol{w}^t - r \frac{\partial \ell_i \left(F(\mathcal{D}_i, \boldsymbol{w}^t \odot \bar{\boldsymbol{m}}^t), y_i \right)}{\partial \boldsymbol{w}^t} \odot \bar{\boldsymbol{m}}_i^t \tag{1}$$

Where $\bar{\boldsymbol{m}}_i^t = \mathsf{P}_i(\boldsymbol{w}_i^t)$, r is the learning rate and $\odot$ is the Hadamard product. The *base* pruning mask $\bar{\boldsymbol{m}}_i^t$ exhibits variability across different base pruning schemes, reflecting the diverse pruning strategy $\mathsf{P}_i(\cdot)$ employed.

Server: The server receives model updates $\hat{\boldsymbol{w}}_i^t$ from all participating users and aggregates to update the global model, as it is typical in FL. We do not consider secure aggregation [33], differential privacy [27], or other defenses known in the literature.[1]

Threat Model. We consider an *honest-but-curious* server, which correctly follows the FL protocol but uses model updates to infer private information, by launching privacy attacks against target user(s), as shown in Algorithm 1. In our case, the server has full visibility of all masked local models and, upon receiving an update from a target user, it attempts to reconstruct the user's training data. We consider reconstruction attacks that invert gradients, including Deep-Leakage-from-Gradients (DLG) [45], Gradient Inversion (GI) [11], and a custom attack SGI described in Section 4.2. We focus on single-round attacks, *i.e.,* attacks that invert gradients observed at a single round[2].

[1]We consider those defenses as out-of-the-scope for this paper, since they are orthogonal to pruning. They can be implemented on top/combined with our pruning-based defense methods.
[2]To the best of our knowledge, there are currently no known *practical* multi-round inverting-the-gradient types of attacks that combine information from multiple rounds. There is theoretical analysis of upper *bounds* of multi-round information leakage, including [9, 32] and we derive our own multi-round

Defense via Pruning: In this paper, we propose to defend against privacy attacks, via pruning itself. One could specifically design an entire pruning scheme not only for accuracy and efficiency but also for privacy, and the defense would then be optimized jointly with the particular base pruning scheme *e.g.,* PruneFL [17] and FedDST [3].

In this paper, we take a modular and universal approach: we consider *any* given state-of-the-art base pruning scheme (with rate $\bar{p}_i$ and mask $\bar{\boldsymbol{m}}_i^t$) without any modification. Then, we introduce additional local *defense* pruning, with mask $\hat{\boldsymbol{m}}_i^t$ and pruning rate $\hat{p}_i$, specifically designed for protecting each user i from privacy attacks. User i eventually sends back to the server weights after applying both base ($\bar{\boldsymbol{m}}_i^t$) and defense ($\hat{\boldsymbol{m}}_i^t$) pruning: $\hat{\boldsymbol{w}}_i^t = \bar{\boldsymbol{m}}_i^t \odot \hat{\boldsymbol{m}}_i^t \odot \boldsymbol{w}_i^t$. This is depicted in Fig. 1 and Algorithm 1 (see Line 15 and 16), and elaborated upon on Fig. 7. Given a base pruning ($\bar{\boldsymbol{m}}_i^t$), there are many possible defense pruning strategies ($\hat{\boldsymbol{m}}_i^t$). In Section 5.2, we propose *PriPrune* for $\hat{\boldsymbol{m}}_i^t$, to optimize the accuracy-vs-privacy tradeoff.

4 QUANTIFYING PRIVACY IN FL MODEL PRUNING

In this section, to address question Q1, we provide a theoretical and empirical quantification of privacy leakage in FL model pruning. We aim to assess the server's ability to infer information about an individual user's local dataset $\mathcal{D}_i$. We employ Mutual Information (MI) as the metric for quantifying privacy leakage:

$$\begin{aligned} \mathsf{I}(X;Y) &= \int_{\mathcal{X}\times\mathcal{Y}} log \frac{d\mathsf{P}_{X,Y}(x,y)}{d\mathsf{P}_X(x) \otimes \mathsf{P}_Y(y)} d\mathsf{P}_{X,Y}(x,y) \\ &= \mathsf{H}(X) + \mathsf{H}(Y) - \mathsf{H}(X,Y) \end{aligned} \tag{2}$$

Where $(X;Y)$ is a pair of random variables with values over the space $\mathcal{X} \times \mathcal{Y}$. Their joint distribution is $\mathsf{P}_{X,Y}(x,y)$ and the marginal distributions are $\mathsf{P}_X(x)$ and $\mathsf{P}_Y(y)$.

This concept, rooted in information theory and Shannon entropy, captures the interdependence between two random variables. It is very powerful for privacy analysis, given its applicability across various domains and tasks [1, 2]. This applicability extends across various datasets, pruning schemes, and attack scenarios, as recently demonstrated in the quantification of privacy within the context of FL aggregation [9].

4.1 Theoretical Quantification

We seek to quantify how much information the server can infer about the private data $\mathcal{D}_i$ of user i over T global training rounds, based on the pruned updates $\left\{\boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t\right\}_{t\in[T]}$ submitted by user i, via:

$$\mathsf{I}_i = \mathsf{I}\left(\mathcal{D}_i;\ \left\{\boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t\right\}_{t\in[T]}\right) \tag{3}$$

where I represents mutual information between $\mathcal{D}_i$ and $\left\{\boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t\right\}_{t\in[T]}$, $\boldsymbol{w}_i^t \in \mathsf{R}^d$ is the local model weights of user i at round t.

This quantification is based on two mild assumptions:

Assumption 1. *The random vector $\hat{g}_i\left(\boldsymbol{w}_i^t, b\right)$ has non-singular covariance matrix Σ_i and mean 0.*

Here, $\hat{g}_i\left(\boldsymbol{w}_i^t, b\right) \in \mathsf{R}^{d^*}$ is the largest sub-vector of the $g_i\left(\boldsymbol{w}_i^t, b\right)$, where d^* is the rank of $g_i\left(\boldsymbol{w}_i^t, b\right)$, $d^* \leq d$, d is the model size, b denotes the size of the random samples. $g_i(\boldsymbol{w}_i^t, b)$ denotes the stochastic estimate of the gradient of the

leakage in Section 4.1.2. Designing a practical multi-round attack is orthogonal to our contribution, which focuses on quantifying the leakage and designing an add-on defense, for *existing* attacks.

local loss function for user i, computed based on a random sample b, and B_i^t is a data batch of size B sampled uniformly at random from it local dataset $\mathcal{D}_i$.

Assumption 2. *$\bar{p}_i$ the pruning rate of user i remains unchanged throughout the training process.*

Building upon Equation 2, Equation 3 can be written as:

$$\mathsf{I}_i \leq \sum_{t=1}^{\tilde{\mathcal{O}}} \mathsf{I}\left(\mathrm{x}_i^t; \boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]} \tag{4}$$

Where $\mathrm{x}_i^t = \frac{\partial \ell_i(\tilde{\boldsymbol{w}}_i^{t-1};\mathcal{D}_i)}{\partial \boldsymbol{w}} = \frac{1}{B}\sum_{b\in\mathsf{B}_i^t} g_i\left(\boldsymbol{w}_i^t, b\right)$.

To that end, we first characterize the privacy leakage for a *single round* t:

$$\mathsf{I}\left(\mathrm{x}_i^t; \boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]} \tag{5}$$

Under these two mild assumptions, we obtain our main theoretical result; the proof is deferred to Appendix A.

4.1.1 Upper Bound for Privacy Leakage in a Single Round. We now provide an upper bound for Equation 5, i.e., for the privacy leakage in a single round.

Theorem 4.1 (Single Round Leakage). *Under Assumption 1 and 2, we have an upper bound of I_i^t for a single round:*

$$\mathsf{I}_i^t \leq 1 - \frac{\bar{p}_i - 1}{2\ln 2} + 2\log\frac{1}{B} + 2\Delta + d^*\log(2\pi e) \tag{6}$$

where $\Delta = \log\det\left(\Sigma_i^{-\frac{1}{2}}\right)$.

Proof. Based on Equation 2, we have

$$\mathsf{I}\left(\mathrm{x}_i^t; \boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]} \leq \underbrace{\mathsf{H}\left(\boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]}}_{\mathsf{A}} + \underbrace{\mathsf{H}\left(\mathrm{x}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]}}_{\mathsf{B}} \tag{7}$$

For part A, based on the chain rule of entropy, we have

$$\mathsf{A} = \mathsf{H}\left(\boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]} + \mathsf{H}\left(\boldsymbol{w}_i^t, \bar{\boldsymbol{m}}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t]} \geq 0 \tag{8}$$

Due to $\mathsf{H}\left(\boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \,\middle|\, \boldsymbol{w}_i^t, \bar{\boldsymbol{m}}_i^t, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]} = 0$, and $\mathsf{H}\left(\boldsymbol{w}_i^t, \bar{\boldsymbol{m}}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t]} \geq 0$, we have

$$\mathsf{H}\left(\boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]} \leq \underbrace{\mathsf{H}\left(\mathrm{x}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]}}_{\mathsf{B}} + 1 - \frac{\bar{p}_i - 1}{2\ln 2} \tag{9}$$

The proof is shown in the Appendix A.

$$\mathsf{B} = \mathsf{H}\left(\frac{1}{B}\sum_{b\in\mathsf{B}_i^t}^{\tilde{\mathcal{O}}} g_i\left(\boldsymbol{w}_i^t, b\right) \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]} \tag{10}$$

Let $X = \sum_{b\in\mathsf{B}_i^t} g_i\ \boldsymbol{w}_i^t, b\ , Z = \left(\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]}$, $Y = \frac{1}{B}X$. $f_{X,Z}(x,z)$ and $f_{Y,Z}(y,z)$ are the corresponding joint probability density functions.

Then Equation 10 can be written as follows:

$$\mathsf{H}\left(Y|Z\right) = \mathsf{H}\Big(\underbrace{\sum_{b\in\mathsf{B}_i^t} g_i\ \boldsymbol{w}_i^t, b}_{\mathsf{A}}\ \Big|\ \underbrace{\left(\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]}}_{\mathsf{C}}\Big) + \log\frac{1}{B} \tag{11}$$

The proof is shown in the Appendix A.

In recent theoretical results for analyzing the behaviour of SGD, the SGD vector is approximated by a distribution with independent components or by a multivariate Gaussian vector [46]. Based on the ZCA whitening transformation and Assumption 1, we have $\hat{g}_i\ \boldsymbol{w}_i^t, b\ = \Sigma_i^{-\frac{1}{2}}\boldsymbol{z}$, where $\boldsymbol{z}$ has zero mean and I_{d^*} convariance matrix.

For the part C, we set $\boldsymbol{v} = \sum_{b\in\mathsf{B}_i^t}\boldsymbol{z}$ and $\boldsymbol{u} = \Sigma_i^{-\frac{1}{2}}\boldsymbol{v}$, then we have:

$$\mathsf{C} = \mathsf{H}\Big(\underbrace{\sum_{b\in\mathsf{B}_i^t} g_i\ \boldsymbol{w}_i^t, b}_{\mathsf{A}}\ \Big|\ \left(\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]}\Big) = \mathsf{H}\Big(\Sigma_i^{-\frac{1}{2}}\underbrace{\sum_{b\in\mathsf{B}_i^t}\boldsymbol{z}}_{\mathsf{A}}\ \Big|\ \left(\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]}\Big) = -\int \mathsf{P}_{\mathcal{U}}(\boldsymbol{u})\log\mathsf{P}_{\mathcal{U}}(\boldsymbol{u})\,\mathrm{d}\boldsymbol{u}$$

$$\overset{(a)}{=} -\int \frac{\mathsf{P}_{\mathcal{V}}(\boldsymbol{v})}{\det\ \Sigma_i^{-\frac{1}{2}}}\log\frac{\mathsf{P}_{\mathcal{V}}(\boldsymbol{v})}{\det\ \Sigma_i^{-\frac{1}{2}}}\det\ \Sigma_i^{-\frac{1}{2}}\ \mathrm{d}\boldsymbol{v} = \log\ \det\ \Sigma_i^{-\frac{1}{2}}\ + \mathsf{H}\Big(\underbrace{\sum_{b\in\mathsf{B}_i^t}\boldsymbol{z}}_{\mathsf{D}}\ \Big|\ \left(\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]}\Big) \tag{12}$$

where (a) is based on the transformation of random vectors. Based on the Maximum Entropy Upper Bound [35], which the continuous distribution X with prescribed variance $\mathcal{V}(\mathrm{X})$ maximizing the entropy is the Gaussian distribution of same variance. Since the entropy of a multivariate Gaussian distribution with mean μ and covariance $K_{i,j}$ is

$$\mathsf{h}(\mathcal{N}_n(\mu,K)) = \frac{1}{2}\log\left(2\pi e\right)^n|K|$$

where $|K|$ denotes the determinant of K. Hence, the maximum entropy upper bound for part D is

$$\mathsf{D} \le \frac{1}{2}\log\left(2\pi e\right)^{d^*}|\mathsf{I}_{d^*}| = \frac{d^*}{2}\log\left(2\pi e\right) \tag{13}$$

Therefore, by combining Equation 12 and Equation 13, the upper bound for part B is

$$\mathsf{B} \le \log\frac{1}{B} + \log\ \det\ \Sigma_i^{-\frac{1}{2}}\ + \frac{d^*}{2}\log\left(2\pi e\right) \tag{14}$$

By adding parts A and B, we establish the upper bound for privacy leakage for a single round as in Theorem 4.1.

4.1.2 Upper Bound for Privacy Leakage across Multiple Rounds. Using Equation 4, we can also obtain an upper bound for Equation 3, i.e., how much information the aggregated model over T global training rounds could leak about the private data:

Algorithm 2 Sparse Gradient Inversion (SGI) Attack

Input: $F(\mathcal{D}_i; \hat{\boldsymbol{w}}_i^t)$: model at round t from targeted user i; learning rate η for inverting gradient optimizer; S: max iterations for attack; τ: regularization term for cosine loss in inverting gradient attack; $\boldsymbol{m}_{rec}$:the recovery mask generated by server; d:the model size
Output: reconstructed training data $(\mathcal{D}_i, y_i)$ at round t

$$\boldsymbol{m}_{rec}[j]_{j\in d} = \begin{cases} 1 & \text{if } \hat{\boldsymbol{w}}_i^t[j] < 0 \\ 0 & \text{if } \hat{\boldsymbol{w}}_i^t[j] = 0 \end{cases}$$

$\nabla \hat{\boldsymbol{w}}_i^t \leftarrow \hat{\boldsymbol{w}}_i^t - \hat{\boldsymbol{w}}_i^{t-1} \odot \boldsymbol{m}_{rec}$
Initialize $\mathcal{D}_0' \leftarrow \mathcal{N}(0,1)$, $y_0' \leftarrow Randint(0, max(y))$
for $s \leftarrow 0$ **to** $S-1$ **do**
  $\nabla \boldsymbol{w}_s' \leftarrow \partial \ell(F(\mathcal{D}_s', \hat{\boldsymbol{w}}_i^t), y_s')/\partial \hat{\boldsymbol{w}}_i^t$
  $\mathcal{L}_s' \leftarrow 1 - \frac{\nabla \hat{\boldsymbol{w}}_i^t \cdot \nabla \boldsymbol{w}_s'}{\|\nabla \hat{\boldsymbol{w}}_i^t\| \|\nabla \boldsymbol{w}_s'\|} + \tau$
  $\mathcal{D}_{s+1}' \leftarrow \mathcal{D}_s' - \eta \nabla_{\mathcal{D}_s'} \mathcal{L}_s'$, $y_{s+1}' \leftarrow y_s' - \eta \nabla_{y_s'} \mathcal{L}_s'$
end for
Return $\mathcal{D}_S', y_S'$

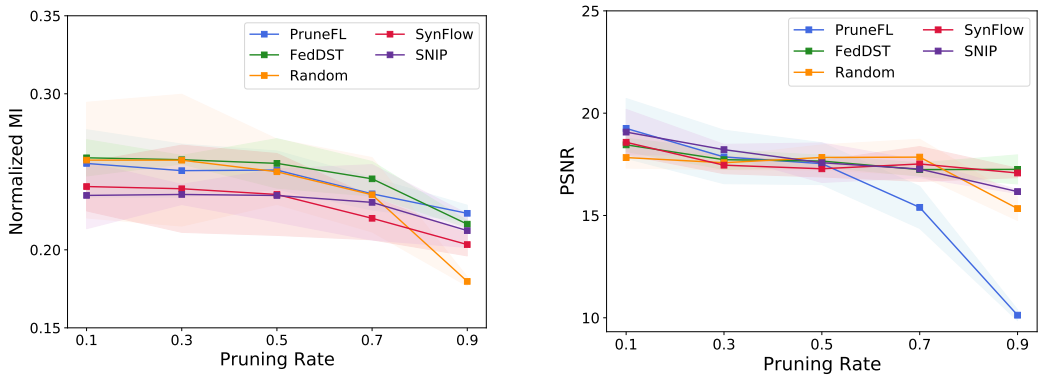
Corollary 4.2 (Multiple Rounds Leakage). *Continuing with Theorem 4.1, we can upper bound* I_i *after* T *global training rounds as follows:*

$$\mathsf{I}_i \leq T \left(1 - \frac{\bar{p}_i - 1}{2\ln 2} + 2\log\frac{1}{B} + 2\Delta + d^* \log(2\pi e)\right) \tag{15}$$

This shows that increasing the number of global training rounds (T) leads to a proportional rise in the upper bound of information leakage from the user's local training model.

Importance of the Upper Bounds: The derived upper bounds are important for several reasons.

- They provide a theoretical upper limit to the amount of information that can potentially be leaked from model updates in pruned models. To the best of our knowledge, this has not been previously characterized for pruned models.
- Because they are information-theoretical in nature, these bounds are universally applicable. They apply to *any* learning task, *any* data distribution, *any* pruning scheme, and *any* privacy attack. The attacker is the server that sees the pruned updates $\boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \;_{t\in[T]}$ from client i, and infers information about the local training Data $\mathcal{D}_i$ of that client. Regardless of the exact reconstruction strategy, the inferred information is at most the mutual information I_i between the update and the local data, quantified in Equation 3. The derivation of the bounds assumes the existence of a pruning mask but does not depend on how this mask was generated, *e.g.,* whether through magnitude-based pruning, gradient-based pruning, or any other pruning method.
- Theorem 1 expresses the bound in terms of important parameters (e.g., the pruning rate p, model size d, etc), which provides insights into the choices and parameters that affect the privacy provided by pruning in practical algorithms. However, the bounds may or may not be tight compared to the actual information leakage, depending on the real-world scenario. In fact, these bounds are necessarily conservative since they are universal. This is why, in the next section, we complement the theoretical analysis with experimental evaluations in specific settings (i.e., considering specific base pruning, attack and defense schemes, specific data sets, and learning tasks).



(a) Normalized Mutual Information (NMI).

(b) Peak Signal-to-Noise Ratio (PSNR).

Fig. 2. Impact of varying pruning rate on privacy leakage using key metrics: Normalized Mutual Information (NMI) displayed on (a) and Peak Signal-to-Noise Ratio (PSNR) shown on (b) under a Sparse Gradient Inversion attack (SGI) in FEMNIST for five pruning methods under varying the base pruning rate.

4.2 Empirical Quantification

We perform an experimental evaluation to quantify the exact amount of privacy loss in concrete settings. This allows us to compare the theoretical bounds to experimental results and show that they are qualitatively aligned. It also allows us to obtain insights (in Section 5.1) that inform the design of defense pruning (in Section 5.2). More evaluation results are provided in Section 6 and in Appendix C.1.

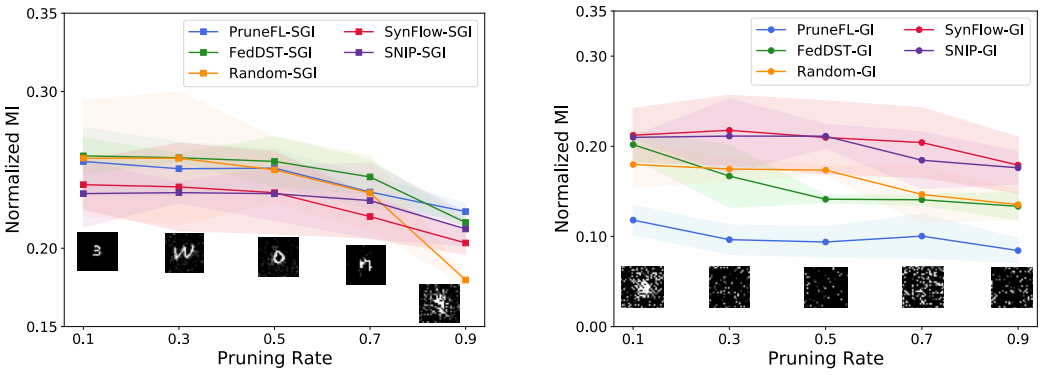
4.2.1 Privacy Metrics. To quantify the extent of privacy leakage, we use the Normalized Mutual Information (NMI) between the training data and the reconstructed data. For two vectors $U, V \in \mathcal{R}^N$, NMI is computed as follows. The entropy of U is calculated by $H(U) = -\sum_{i=1}^{|U|} P(i)\log(P(i))$, where $P(i) = |U_i|/N$. The Mutual Information between U and V is given by $I(U;V) = \sum_{i=1}^{|U|}\sum_{j=1}^{|V|} \frac{|U_i \cap V_j|}{N} \log \frac{N|U_i \cap V_j|}{|U_i||V_j|}$. NMI is then measured as $\frac{I(U;V)}{\text{mean}(H(U),H(V))}$.

For image data, prior research [9] has shown that NMI aligns well with the Peak Signal-to-Noise Ratio (PSNR). This is also the case here: we see an agreement between NMI and PSNR with increasing pruning rates in Fig. 2 and Fig. A2, both showing a similar decreasing trend. In the rest of the paper, we use both NMI and PSNR as metrics to quantify the success of a reconstruction attack. Intuitively, the higher the NMI, the higher the privacy leakage, the more successful the reconstruction attack. The higher the PSNR, the closer the original and reconstructed images and the more successful the attack. Hence, Fig. 2 and Fig. A2 compare the NMI and PSNR metrics and show that that increasing pruning rate reduces the success of the privacy attack.

4.2.2 Reconstruction Attack Algorithms. We first implement the classic Gradient Inversion (GI) attack [11] from the DLG attack family. However, due to the unique sparsity patterns introduced by model pruning in FL, we identify the potential for further optimizing this attack to specifically exploit the sparsity. Consequently, we introduce an advanced attack tailored for the context of model pruning in FL, the Sparse Gradient Inversion (SGI) attack. This attack is designed to recover the pruning mask within the pruned model. Moreover, its adaptability allows seamless integration with existing privacy attacks in FL, thereby amplifying their effectiveness.

Method	Pruning Criteria	Pruning Cycles	Pruning at Server	Pruning at Client
SNIP [21]	gradient-based	one shot	✓	✗
SynFlow [34]	gradient-based	one shot	✓	✗
GraSP [38]	gradient-based	one shot	✓	✗
Random Pruning	magnitude-based	iterative	✓	✓
FedDST [3]	magnitude-based	iterative	✓	✓
PruneFL [17]	magnitude-based	iterative	✓	✓

Table 1. Comparison for Evaluated Methods

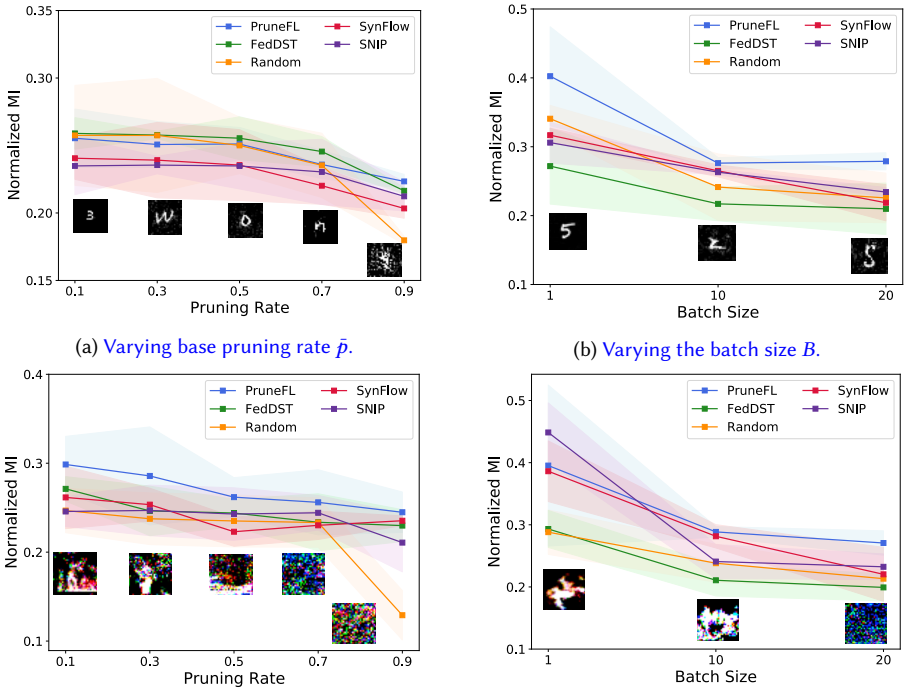


(a) SGI attack. (b) GI attack.

Fig. 3. Comparison between the SGI (Sparse Gradient Inversion) attack and the GI (Gradient Inversion) attack on the FEMNIST dataset for five pruning methods under varying the base pruning rate. In addition to the NMI metric, we show the corresponding recovered images, which show the degradation with an increasing pruning rate.

The SGI attack is provided in Algorithm 2. In each iteration t, the SGI algorithm proceeds as follows: (i) Based on the user i' local model weights $\boldsymbol{w}_i^t$, the server attempts to recover the pruning mask of user i (Line 2). The recovered mask is then integrated into the calculation of gradient, allowing the server to obtain a gradient $\nabla\hat{\boldsymbol{w}}_i^t$ that closely approximates the true gradients of user i; (ii) The SGI attack randomly initializes a set of dummy data, comprising dummy inputs $\mathcal{D}_0'$ and dummy labels y_0'. (iii) After the dummy gradient $\boldsymbol{w}'$ is acquired, the server then updates the dummy data in the direction that minimizes the cosine distance between the dummy gradient and the gradient $\nabla\hat{\boldsymbol{w}}_i^t$.

4.2.3 Base Pruning Schemes under Attack. We subject a set of well-established FL pruning methods to the privacy attack to assess their vulnerabilities and effectiveness in protecting user privacy. The pruning methods under evaluation include Random pruning, Snip [21], SynFlow [34], GraSP [38], FedDST [3], and PruneFL [17], all of which are discussed in Related Work (Section 2). We present an overview of the pruning criteria, cycles, and schedules employed by the six evaluated methods. These specifics are elaborated in Table 1. By scrutinizing the table, we discern notable trends: three of the methods execute pruning exclusively at the server side, while the remaining methods engage in pruning activities at both the server and client sides. The latter approaches adopt an iterative pruning strategy throughout the course of the FL process.



(a) Varying base pruning rate $\bar{p}$.

(b) Varying the batch size B.

(c) Varying base pruning rate $\bar{p}$.

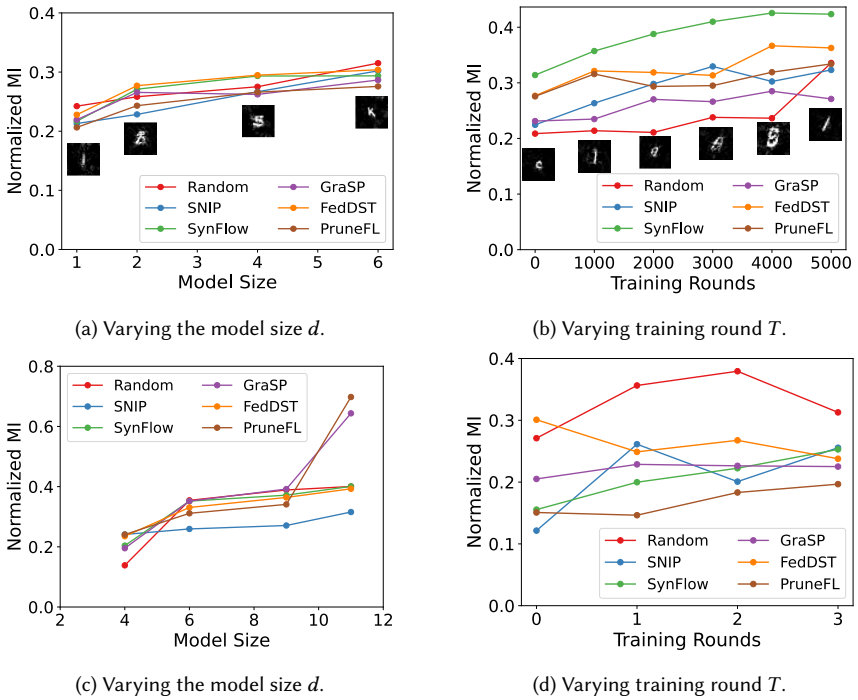
(d) Varying the batch size B.

Fig. 4. Impact of varying base pruning rate $\bar{p}$ and batch size B on privacy leakage. We consider the Normalized Mutual Information (NMI) as privacy metric. We launch a Sparse Gradient Inversion (SGI) attack against 5 base pruning methods (Random, SNIP, SynFlow, FedDST, and PruneFL), over two datasets (FEMNIST in the First Row and CIFAR10 in the (Second Row).

We compare the effectiveness of the Sparse Gradient Inversion (SGI) attack with the Gradient Inversion attack under varying the base pruning rate. Our evaluation is based on the normalized mutual information (NMI) metric. Specifically, we analyze the NMI achieved by each attack for different pruning methods. Figure 3 demonstrates that the SGI attack consistently outperforms the Gradient Inversion attack by inferring a larger amount of information (as captured by NMI) at each pruning rate level, across all tested pruning methods. This is because the SGI attack exploits the sparsity inherent to model pruning, ultimately allowing better reconstruction attacks. In the rest of the paper, we use the SGI attack as a baseline, with a learning rate of 0.01 and 10,000 attack iterations.

4.2.4 Attack Performance. We launch the SGI attack on the aforementioned FL pruning schemes, on the FEMNIST and CIFAR10 datasets, to assess how FL parameters influence privacy leakage. The experimental results in Fig. 4 and Fig. 5 demonstrate that the impact of the parameters $\bar{p}$, B, d, and T qualitatively align with our theoretical findings.

Impact of Pruning Rate ($\bar{p}$). Fig. 4a and Fig. 4c illustrate that a higher pruning rate generally leads to a decrease in NMI. This is intuitively expected as pruning removes information and makes it more difficult to attack the original model. The effect is clearer for large pruning rates (*e.g.*, above 0.5). Image reconstruction is clearer in FEMNIST than CIFAR, consistently with prior work.



(a) Varying the model size d.

(b) Varying training round T.

(c) Varying the model size d.

(d) Varying training round T.

Fig. 5. Impact of varying model size d and training rounds T on privacy leakage using Normalized Mutual Information (NMI) under a Sparse Gradient Inversion (SGI) attack for six pruning methods: Random, SNIP, SynFlow, GraSP, FedDST, and PruneFL in FEMNIST dataset (First Row) and CIFAR10 dataset (Second Row).

Impact of Batch Size (B). Fig. 4b and Fig. 4d show increasing the batch size B contributes to reducing information leakage from the local training model. Larger batch sizes enable more data to be added during model training, which can help obscure single data and increase privacy protection.

Impact of Model Size (d). We consider 4 architecture of the models for FEMNIST dataset: Conv-1, Conv-2, Conv-4, and Conv-6, featuring 1, 2, 4, and 6 convolutional layers, along with 2 linear layers. These models encompass 3,815,566, 6,603,710, 13,657,406, and 25,324,350 parameters, respectively. We consider 4 different model architectures for CIFAR 10 dataset: VGG-4, VGG-6, VGG-9, and VGG-11. These models consist of 1 convolutional layer and 2 linear layers, 4 convolutional layers and 2 linear layers, 6 convolutional layers with 3 linear layers, and 8 convolutional layers with 3 linear layers, respectively. These models encompass 1,125,642, 1,814,538, 6,436,106, and 9,385,994 parameters, respectively. Figure 5a and Figure 5c depicts the relationship between the model size (d) and the corresponding information leakage. The x-axis in the figure represents the number of layers in each model. Fig. 5a and Fig. 5c show the upper bounds on information leakage increase as d^* increases. Interestingly, the impact of model size d on information leakage is not linear. In the case of over-parameterised models, some parameters may not contribute to the model's performance or information retention. Therefore, increasing the size of such models may not proportionally affect information leakage.

Impact of Global Training Rounds (T). Fig. 5b and Fig. 5d show that increasing the global training rounds (T) leads to a proportional rise in the upper bound of information leakage. As the training progresses over multiple rounds, the

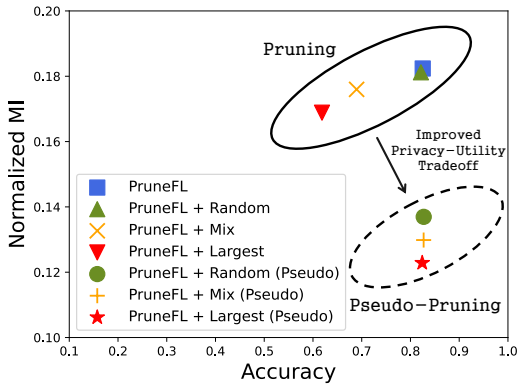


Fig. 6. Tradeoff of privacy vs. accuracy with three defense strategies: *Largest* (red), *Random* (green), and *Mix* (yellow), and their enhanced versions incorporating *Pseudo-Pruning* (*Pseudo*). *Pseudo* could improve the overall tradeoff by elevating model accuracy and *Largest* offers the highest level of privacy protection among all defense methods.

user's model updates are repeatedly exposed to the central server. This repeated exposure increases the risk of potential memorization of private training information by the server.

5 DESIGN OF DEFENSE PRUNING

5.1 Insights from Privacy Quantification

As shown above, pruning designed only with model accuracy and model size in mind, does not sufficiently protect against privacy attacks. Next, we describe three insights gained in the process of privacy quantification that inform our design of privacy-aware FL model pruning (*PriPrune* described in Section 5.2).

Insight 1: Largest Weights Matter. Pruning weights with large gradients improves privacy the most, but also hurts model accuracy the most. The intuition is that (base) pruning criteria, as used *e.g.*, in *PruneFL* and *FedDST*, avoid pruning model weights with the largest gradients, in order to preserve the most valuable information for the model. These gradients can then be exploited by gradient inversion attacks. Fig. 4a and Fig. 4c confirm empirically that, as more weights with large gradients are pruned, the attack is less successful.

This implies that the defense should strategically prune certain weights with large gradients, in addition to those pruned by the base pruning scheme. We explored three defense strategies for weights to prune: (1) weights with the top-k largest gradients (denoted as *Largest*), (2) random weight pruning (denoted as *Random*), and (3) a hybrid approach combining the *Largest* and *Random* (denoted as *Mix*). We compared the three strategies for the same defense pruning rate $\hat{p}_i$ (set 0.3) on top of the base method (*PruneFL*) and details are provided in Appendix D.1. The results in Fig. 6, within the Pruning oval, show the tradeoff achieved between privacy and accuracy: the "Largest" method achieves the best privacy and the worst model accuracy. This is because pruning weights with large gradients, removes valuable information, thus improving privacy but also significantly reducing model accuracy. Therefore, to improve this tradeoff and preserve privacy without harming accuracy, we need to introduce an additional mechanism.

Insight 2: *Pseudo-Pruning*. Users can prune weights with large magnitude gradients when they communicate with the server (which helps privacy) *and* still keep these weights for their local training (which maintains accuracy).

We refer to this idea as *Pseudo-Pruning*. and illustrate it in Fig. 7. Before sending the weights $\bar{\boldsymbol{w}}_i^t$ to the server user i conducts *Pseudo-Pruning* with defense mask $\hat{\boldsymbol{m}}_i^t$. The weights pruned out by defense pruning with large gradients $\tilde{\boldsymbol{w}}_i^t$

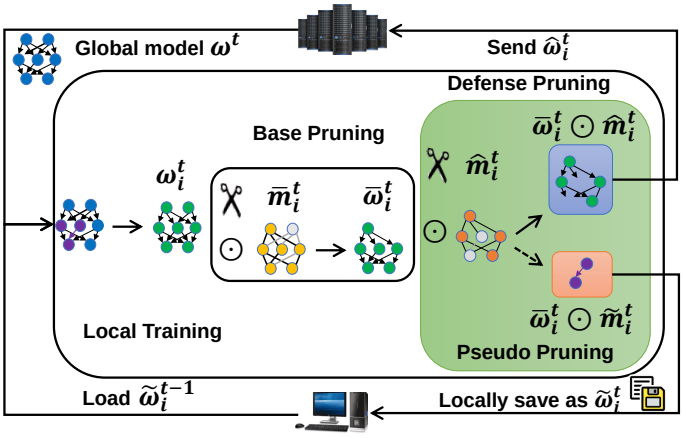


Fig. 7. Illustration of local pruning at user i with *Pseudo-Pruning* defense: First user applies base pruning with base mask $\bar{m}_i$ to derive a pruned model $\bar{w}_i^t$. Then, defense pruning is conducted with mask $\hat{m}_i$. Defense pruning is implemented as *Pseudo-Pruning*, where the remaining weights $\hat{w}_i^t$ ($= \bar{w}_i^t \odot \hat{m}_i^t$) are sent to the server; while the pruned weights $\tilde{w}_i^t$ ($= \bar{w}_i^t \odot \tilde{m}_i^t$) are locally saved for future local training. Notably, these weights pruned by defense are not actually pruned, hence the term "Pseudo-Pruning."

($= \bar{w}_i^t \odot \tilde{m}_i^t$) are withheld locally, thus preserving accuracy. Meanwhile, the weights remaining after the defense pruning without large gradients, $\hat{w}_i^t$ ($= \bar{w}_i^t \odot \hat{m}_i^t$), are transmitted to the server, thus preserving privacy. Here, $\tilde{m}$ is the bit-wise complement matrix of $\hat{m}$. In the next round, after user i receives the global model w^{t+1}, the locally saved weights $\tilde{w}_i^t$ are loaded into the global model, serving as the local initial model.

We combine *Pseudo-Pruning* with each of the three strategies (*Largest*, *Random*, and *Mix*) in Insight 1, and we show the evaluation results within the Pseudo-Pruning dashed oval in Fig. 6. Fig. 6 shows that the model accuracy is notably better with *Pseudo-Pruning* than real pruning, with the same mask. Furthermore, combining the two insights into *Largest (Pseudo)* offers the most effective privacy guarantee among all defenses discussed above. Hence, we adopt *Largest (Pseudo)* as our defense strategy. Additional evaluation details are available in the Appendix D. Still, a remaining shortcoming of the defense considered so far, is that their pruning rate is manually selected and remains fixed.

Insight 3: Adapt. The defense pruning rate ($\hat{p}_i$) should adapt to the model weights, to jointly optimize privacy and model accuracy. A fixed defense rate is unable to capture the evolving dynamics of the training process. The defense strategy should adapt to the changing dynamics of the training process by employing an adaptive defense pruning rate, enabling a more effective optimization of the tradeoff between privacy and accuracy.

5.2 The *PriPrune* Mechanism

Combining all aforementioned insights, we introduce the defense strategy, PriPrune, which dynamically adapts the *Pseudo-Pruning* defense mask $\hat{m}$ to jointly optimize accuracy and privacy throughout the FL training process.

Loss Function. The objective of user i is to minimize a loss function that combines privacy and accuracy:

$$\mathcal{L}_{local} = \lambda_{acc}\mathcal{L}_{acc} + \lambda_{pri}\mathcal{L}_{pri} + \lambda_{sha} \tilde{\sum_{l\in L, j\in d_l}} \alpha_{i_{(l,j)}}^t \tag{16}$$

Here, $\mathcal{L}_{acc}$ represents the local model training loss, $\mathcal{L}_{pri}$ represents the local privacy loss, and $\hat{\sum_{l\in L, j\in d_l}} \alpha_{i_{(l,j)}}^t$ serves as the privacy regularization term, where $\alpha_{i_{(l,j)}}^t$ represents the probability of user i not sharing the j-th parameter in l-th

layer with the server at round t. L represents the total number of layers in the model, and d_l refers to the number of parameters in the l-th layer. Additionally, λ_{acc}, λ_{pri}, and λ_{sha} act as weights for accuracy loss, privacy loss, and the privacy regularization term, respectively.

Next, we explain the terms of the loss function of the user in Equation 16 and their rationale. First, the accuracy loss is given by: $\mathcal{L}_{acc} = \mathcal{L}\ \mathcal{D}_i; \hat{\boldsymbol{w}}_i^t | \boldsymbol{w}_{i,init}^t\ $. The initial local weight is computed as: $\boldsymbol{w}_{i,init}^t = \boldsymbol{w}^t \odot \hat{\boldsymbol{m}}_i^{t-1} + \bar{\boldsymbol{w}}_i^{t-1} \odot \tilde{\boldsymbol{m}}_i^{t-1}$, where $\tilde{\boldsymbol{m}}$ denotes the bit-wise complement matrix of $\hat{\boldsymbol{m}}$. It is the composition of locally saved weights with global weights through the utilization of the defense mask $\hat{\boldsymbol{m}}$.

Then, the privacy loss $\mathcal{L}_{pri}$ is given by:

$$\mathcal{L}_{pri} = \tilde{\mathrm{O}}_{l \in L, j \in d_l} - \left(\frac{N_l}{N_l} \right)_{l \in L} \left(\frac{|\boldsymbol{g}_{l,j}|}{|\boldsymbol{g}_{l,j}|} \right)_{j \in d_l} \log \boldsymbol{\alpha}_{i_{(l,j)}}^t \tag{17}$$

N_l indicates the number of weights in l-th layer, $|\boldsymbol{g}_{l,j}|$ represents the magnitude of the gradient for the j-th parameter in l-th layer. The privacy loss is designed to discourage sharing weights with large gradients inspired by Insight 1, aiming to mitigate information leakage. Therefore, we assign higher weights to parameters with larger gradients in the loss function. Through this process, during the optimization of the privacy loss, the algorithm updates a higher $\boldsymbol{\alpha}_i^t$ associated with the larger weight term. This higher $\boldsymbol{\alpha}_i^t$ signifies a higher probability of not sharing the parameters with larger gradients with the server, thus promoting the sharing of less valuable information with the server. Moreover, we also distribute constraints to l-th layer based on its parameter count N_l, in proportion to the total model weights, which ensures uniform layer-wise sparsity.

The term $\left| \right|_{l \in L, j \in d_l} \boldsymbol{\alpha}_{i_{(l,j)}}^t$ introduces a privacy penalty based on the magnitude of $\boldsymbol{\alpha}_i^t$. The optimization process aims to minimize the sum of $\boldsymbol{\alpha}_i^t$, resulting in a decrease in the overall probability of not sharing parameters. This, in turn, leads to an increase in the total probability of sharing parameters. Thus, while the privacy loss discourages the sharing of parameters with larger gradients with the server, this term encourages the sharing of parameters with smaller gradients, thus preserving accuracy while maintaining privacy.

Optimization. In *PriPrune*, we optimize the defense mask $\hat{\boldsymbol{m}}$ and model weights $\bar{\boldsymbol{w}}_i$ jointly through gradient descent based on our designed loss function. To overcome the discrete and non-differentiable nature of the defense mask $\hat{\boldsymbol{m}}$, we employ Gumbel-Softmax sampling [16, 25] to substitute the original non-differentiable sample with a differentiable sample. Specifically, $\hat{\boldsymbol{m}}$ is parameterized by a distribution vector $\boldsymbol{\pi}_{l,j} = [\boldsymbol{\alpha}_{i_{(l,j)}}^t, 1 - \boldsymbol{\alpha}_{i_{(l,j)}}^t]$. $v_{l,j}$ represents the soft *Pseudo-Pruning* decision for the j-th parameter in layer l. The reparameterization trick is used for differentiable training:

$$\boldsymbol{v}_{l,j}(k) = \left(\frac{\exp\ (\log \boldsymbol{\pi}_{l,j}(k) + G_{l,j}(k))/\tau}{\exp\ (\log \boldsymbol{\pi}_{l,j}(k) + G_{l,j}(k))/\tau} \right)_{k \in \{0,1\}} \tag{18}$$

where $G_{l,j} = -\log\ -\log U_{l,j}\ $ is Gumbel distribution with $U_{l,j}$ sampled from a uniform i.i.d. distribution Unif$(0, 1)$, τ is the temperature of the softmax and $k \in \{0, 1\}$. We apply the hard sample trick introduced by PyTorch document [29] to acquire the defense mask $\hat{\boldsymbol{m}}$ and the main trick is to do $y_{hard} - f_{StopGradient}(y_{soft}) + y_{soft}$.

PriPrune. The *PriPrune* algorithm is descibed in Algorithm 3, which updates the *UserUpdate* function of Algorithm 1. As depicted in Algorithm 3, in each FL round, after user i receives the latest global model $\boldsymbol{w}_i^t$ from the server, they sample their defense mask $\hat{\boldsymbol{m}}_i$ using Equation 18. The defense mask $\hat{\boldsymbol{m}}_i$ is then employed to combine the global model $\boldsymbol{w}^t$ and the user's locally saved $\tilde{\boldsymbol{w}}_i^{t-1}$ to generate the local initial model $\boldsymbol{w}_{i,init}^t$. Following this composition step, the

Algorithm 3 *PriPrune*

function *UserUpdate* $(\boldsymbol{w}^t, \boldsymbol{m}^t, i)$:
Load $\tilde{\boldsymbol{w}}_i^t$ *from local storage of user* i
Initialize $\boldsymbol{\alpha}_i^t = \boldsymbol{\alpha}_i^{t-1}$
for local epoch $e \in E$ *do*
  Sample defense mask $\hat{\boldsymbol{m}}_i^t$ *by Equation 18 with* $\boldsymbol{\alpha}_i^t$
  Combine local and global model by:

$$\boldsymbol{w}_i^t[j]_{j\in d} = \begin{cases} \tilde{\boldsymbol{w}}_i^t[j] & \text{if } \hat{\boldsymbol{m}}_i^t[j] = 0 \\ \boldsymbol{w}_i^t[j] & \text{if } \hat{\boldsymbol{m}}_i^t[j] = 1 \end{cases}$$

  Base Pruning: $\bar{\boldsymbol{w}}_i^t \leftarrow \boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t$
  $\mathcal{L}_{local} \leftarrow Equation 16.$
  Model parameter update:
  $\bar{\boldsymbol{w}}_i^t \leftarrow \bar{\boldsymbol{w}}_i^t - \eta \nabla \mathcal{L}_{local}(\bar{\boldsymbol{w}}_i^t, \boldsymbol{\alpha}_i^t)$
  Defense mask parameter update:
  $\boldsymbol{\alpha}_i^t \leftarrow \boldsymbol{\alpha}_i^t - \eta \nabla \mathcal{L}_{local}(\bar{\boldsymbol{w}}_i^t, \boldsymbol{\alpha}_i^t)$
end for
$\hat{\boldsymbol{m}}_i^t \leftarrow argmax(\boldsymbol{\alpha}_i^t, 1 - \boldsymbol{\alpha}_i^t)$
Defense Pruning: $\hat{\boldsymbol{w}}_i^t \leftarrow \bar{\boldsymbol{w}}_i^t \odot \hat{\boldsymbol{m}}_i^t, \tilde{\boldsymbol{w}}_i^t \leftarrow \bar{\boldsymbol{w}}_i^t \odot \tilde{\boldsymbol{m}}_i^t$
Return $\hat{\boldsymbol{w}}_i^t$ *to server, locally save* $\tilde{\boldsymbol{w}}_i^t$

user initiates local training based on Equation 16, leading to updates in $\bar{\boldsymbol{w}}_i^t$, $\tilde{\boldsymbol{w}}_i^t$, and $\boldsymbol{\alpha}_i^t$. Then user i conducts defense pruning, retaining the non-shared weights $\tilde{\boldsymbol{w}}_i^t$ locally, and transmits the shared weights $\hat{\boldsymbol{w}}_i^t$ to the server.

6 EXPERIMENTAL EVALUATION

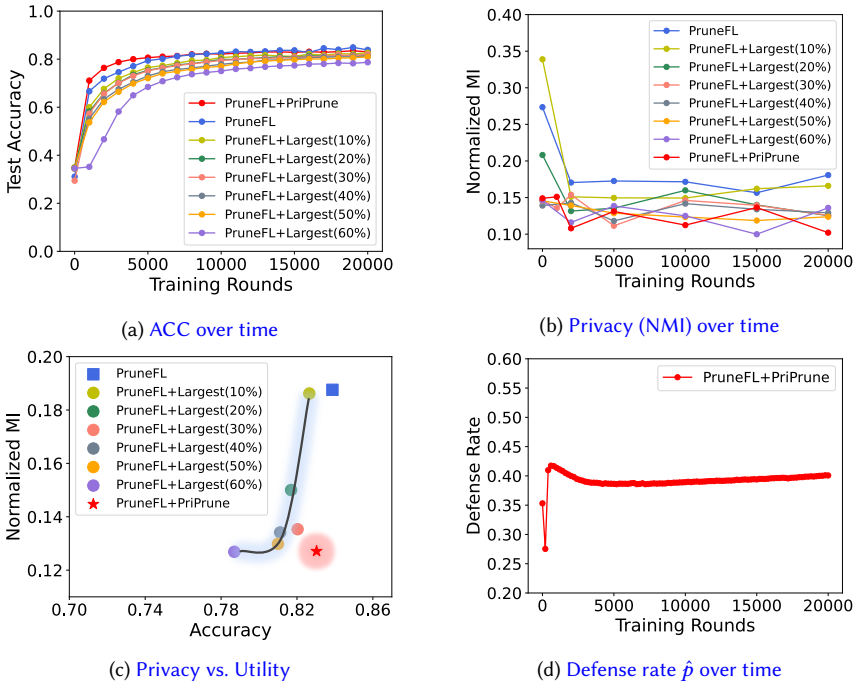
6.1 Experimental Setup

Datasets and Models. We evaluate models Conv-2 [5] and VGG-11 [31]) along with their corresponding datasets FEMNIST [5] and CIFAR-10 [19]), which are commonly employed in FL studies.

Implementation details. In each round of training, we randomly select 10 clients from a pool of 193 users for the FEMNIST and 100 clients for the CIFAR-10. The training process involves 20,000 iterations, with a batch size of 20 for the FEMNIST and a batch size of 1 for the CIFAR-10. We employ 1 local training and initialize the learning rate at 0.25 with the base pruning rate set to 0.3. The details regarding the data split, resource and configurations are provided in Appendix B.1.

Performance Metrics. To assess the effectiveness of the evaluated pruning methods, we employ the privacy metrics and utility metrics. For privacy assessment, we use the Normalized Mutual Information (NMI) between the training data and the reconstructed data as the privacy metric. Higher values of NMI indicate a higher risk of privacy leakage. Additionally, the Peak Signal-to-Noise Ratio (PSNR) is employed as a well-established metric for image quality evaluation. A thorough comparison of NMI and PSNR, along with supplementary experiments, supplemental experiments can be found in Fig. 2 and Appendix B.3. The utility metric relies on model accuracy (ACC) to gauge model performance. Ideally, we want improved privacy, without significantly compromising accuracy.

Hyper-parameter Selection. We conducted a search for our main parameter, the defense rate $\hat{p}_i$, from 0.1 to 0.6, as depicted in Fig. 8c. With regards to our hyperparameters in loss functions, we search λ_{acc} in the list of [1, 5, 10], λ_{pri} in the list of [1, 10, 15] and λ_{sha} in the list of [2e-06, 2e-05, 2e-04]. Detailed results are provided in Appendix D.3.



(a) ACC over time

(b) Privacy (NMI) over time

(c) Privacy vs. Utility

(d) Defense rate $\hat{p}$ over time

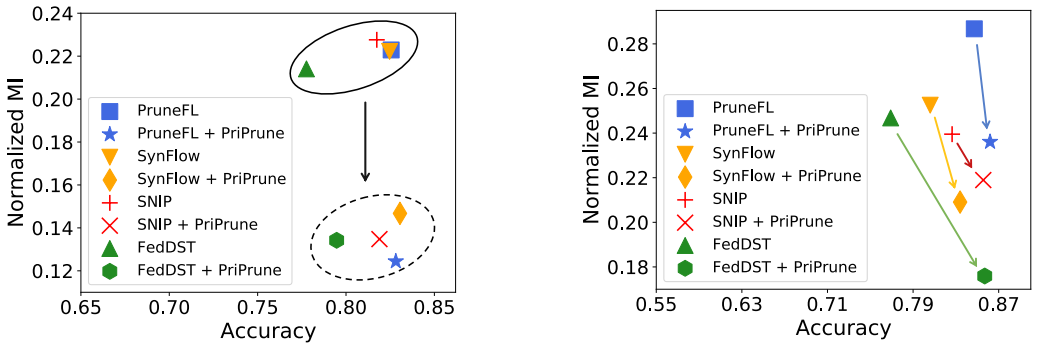
Fig. 8. Performance evaluation of *PriPrune* with adaptive defense rate and *Largest (Pseudo)* with 6 fixed defense rates, as defense strategies, integrated with PruneFL (as the base pruning method), conducted over 20,000 training rounds across the FEMNIST dataset. (a) Comparison of ACC over training rounds under these different defense strategies. (b) Comparison of Privacy (NMI) over training rounds under these different defense strategies. (c) The Privacy-Utility tradeoff, measured in terms of NMI and ACC, respectively.(d) Change in the adaptive defense rate of *PriPrune* over training rounds.

6.2 Performance of *PriPrune*

Recall that in Section 5.1, Insight 1, we initially assessed three defense strategies atop the base pruning method, namely: *Random*, *Largest, Random and Mix*. The results in Figure 6 indicate that while these defense strategies enhance privacy, they also degrade utility. Consequently, in Insight 2, we introduce *Pseudo-Pruning* to optimize this tradeoff. We evaluate the aforementioned three defense strategies both with and without *Pseudo-Pruning*, revealing that their integration with *Pseudo-Pruning* leads to improved accuracy and enhanced privacy. Notably, the *Largest + Pseudo-Pruning* strategy achieves the optimal balance between privacy and utility, thus emerging as our preferred defense mechanism. In Insight 3, we tackle the challenge of a fixed defense rate by proposing *PriPrune*, which incorporates *Largest + Pseudo-Pruning* (*Largest (Pseudo)*) with an adaptive defense rate. The evaluation of the privacy improvement achieved by each individual insight, in Section 5.1, essentially provides an ablation study. In this section, we conduct comprehensive experiments to meticulously evaluate the performance of the entire *PriPrune* across various pruning mechanisms and datasets.

First, we present a performance comparison between the *Largest (Pseudo)* with 6 fixed defense rates and *PriPrune* with the adaptive defense rate, both are integrated with PruneFL (as the base pruning method).

Comparable Utility. Fig. 8a illustrates how the utility metrics, ACC, change over FL rounds. It shows the convergence curve of *PriPrune*, with its model accuracy closely aligned with that of PruneFL. This alignment indicates that *PriPrune*



(a) *PriPrune* in FEMNIST (b) *PriPrune* in CIFAR-10

Fig. 9. Comparison of tradeoffs between privacy-vs-accuracy among four base pruning mechanisms, both with and without the integration of *PriPrune*, across the FEMNIST and CIFAR-10 datasets.

maintains utility comparable to PruneFL. However, for the *Largest (Pseudo)* defense, increasing its fixed defense rate from 10% to 60% results in decreased utility, rendering it unable to match the ACC achieved by PruneFL.

Enhanced Privacy. Figure 8b illustrates the change in the privacy metric NMI across FL rounds. It shows that *PriPrune* consistently maintains a low NMI throughout all FL training rounds, significantly outperforming PruneFL. Moreover, its NMI is comparable to that of *Largest (Pseudo)* with a fixed defense rate of 60%.

Optimal Tradeoff. Figure 8c illustrates the comparison of the privacy and utility tradeoffs among the mentioned defenses. It shows that increasing the defense rate of *Largest (Pseudo)* enhances privacy protection but at the expense of model accuracy. Conversely, *PriPrune* achieves the optimal tradeoff between privacy and accuracy. This is attributed to the adaptive nature of *PriPrune*'s defense rate $\hat{p}$, which adjusts to the model weights and is optimized jointly for privacy and model accuracy. The dynamic change of the defense rate of *PriPrune* is depicted in Fig. 8d.

We have demonstrated that *PriPrune* stands out as the best defense solution among other strategies. Subsequently, we present the privacy performance of *PriPrune* when integrated with various state-of-the-art base pruning schemes in FL, such as SNIP, SynFlow, FedDST, and PruneFL, across both the FEMNIST and CIFAR-10 datasets.

***PriPrune* with State-of-the-Art Pruned FL.** Figure 9 shows the results of using *PriPrune* for defense after a number of state-of-the-art base pruning schemes. Across all base pruning methods, this integration consistently enhances privacy while preserving accuracy. This highlight *PriPrune*'s effectiveness and versatility in achieving an optimal tradeoff between privacy and accuracy in diverse scenarios. This outcome is primarily due to the nature of *PriPrune*, which not only actually prunes but also selectively retains certain critical weights for local training (through pseudo-pruning). This mechanism allows the model to maintain or even improve accuracy while reducing the information leakage that could be exploited in privacy attacks, as also shown in Fig.6. Specifically, *PriPrune* focuses on pseudo-pruning weights with large gradients that are most likely to leak sensitive information and keep them for local training. This selective pruning can lead to a model that is more robust against privacy attacks without a significant loss in accuracy. Additionally, the adaptive pruning rate in *PriPrune* is optimized for the tradeoff between privacy and accuracy, allowing the model to balance these two objectives effectively.

While Fig. 9 compares the tradeoffs between privacy-vs-accuracy, Table 2 further elaborates on the privacy improvement achieved by using *PriPrune* as defense after any base scheme, reporting results over three experiments. Compared to the base approach alone, adding *PriPrune* as defense consistently reduces NMI, thus improving privacy. For instance,

	FEMNIST		CIFAR-10	
Methods	Base	Base + *PriPrune*	Base	Base + *PriPrune*
PruneFL	0.22 ± 0.008	**0.12** ± 0.007	0.29 ± 0.009	**0.24** ± 0.009
FedDST	0.21 ± 0.025	**0.13** ± 0.007	0.25 ± 0.008	**0.18** ± 0.006
SynFlow	0.22 ± 0.009	**0.15** ± 0.005	0.25 ± 0.012	**0.21** ± 0.018
SNIP	0.23 ± 0.012	**0.13** ± 0.014	0.24 ± 0.004	**0.22** ± 0.008

Table 2. Comparison of privacy levels (NMI) between Base and Base + *PriPrune* for FL pruning methods. A lower NMI indicates a reduced risk of privacy leakage.

on the FEMNIST dataset, *PriPrune* improves the privacy of *PruneFL* by 45.5% without compromising accuracy; on the CIFAR-10 dataset, *PriPrune* improves privacy of *FedDST* by 28% while maintaining accuracy.

7 CONCLUSION

In this paper, we revisit federated learning with model pruning, from the point of view of privacy. First, we quantify the information leakage and privacy gain offered for model pruning in FL, both theoretically and experimentally. Inspired by insights obtained from our extensive privacy quantification, we design *PriPrune*– an adaptive privacy-preserving local pruning mechanism in FL, that is jointly optimized for privacy and model performance. Our proposed mechanism achieves the best tradeoff between privacy and accuracy across diverse pruning methods and datasets under privacy attacks. One direction for our future work is to combine our pruning-based defense with classic, orthogonal defenses in FL such as differential privacy and secure aggregation, to enhance privacy further.

ACKNOWLEDGMENT

Tianyue Chu and Nikolaos Laoutaris were supported by the MLEDGE project (REGAGE22e00052829516), funded by the Ministry of Economic Affairs and Digital Transformation and the European Union-NextGenerationEU/PRTR. Mengwei Yang and Athina Markopoulou were supported by NSF awards 1956393 and 1900654.

REFERENCES

[1] Philip Bachman, R Devon Hjelm, and William Buchwalter. 2019. Learning representations by maximizing mutual information across views. *Advances in neural information processing systems* 32 (2019).

[2] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. 2018. Mutual information neural estimation. In *International Conference on Machine Learning(ICML)*. PMLR, 531–540.

[3] Sameer Bibikar, Haris Vikalo, Zhangyang Wang, and Xiaohan Chen. 2022. Federated dynamic sparse training: Computing less, communicating less, yet learning better. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 6080–6088.

[4] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. 2017. Practical Secure Aggregation for Privacy-Preserving Machine Learning *(CCS '17)*. Association for Computing Machinery, New York, NY, USA, 1175–1191. https://doi.org/10.1145/3133956.3133982

[5] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. 2019. Leaf: A benchmark for federated settings. In *33rd Conference on Neural Information Processing Systems (NeurIPS)*.

[6] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. 2020. The lottery ticket hypothesis for pre-trained bert networks. *Advances in neural information processing systems(NeurIPS)* 33 (2020), 15834–15846.

[7] Min Du, Ruoxi Jia, and Dawn Song. 2020. Robust anomaly detection and backdoor attack detection via differential privacy. In *International Conference on Learning Representations(ICLR)*.

[8] Cynthia Dwork. 2006. Differential privacy. In *Automata, Languages and Programming: 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II 33*. Springer, Berlin, Heidelberg, 1–12.

[9] Ahmed Roushdy Elkordy, Jiang Zhang, Yahya H Ezzeldin, Konstantinos Psounis, and Salman Avestimehr. 2023. How Much Privacy Does Federated Learning with Secure Aggregation Guarantee?. In *Proceedings on Privacy Enhancing Technologies (PoPETs)*.

[10] Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations(ICLR)*.

[11] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. 2020. Inverting gradients-how easy is it to break privacy in federated learning? *Advances in Neural Information Processing Systems(NeurIPS)* 33 (2020), 16937–16947.

[12] Antonious Girgis, Deepesh Data, Suhas Diggavi, Peter Kairouz, and Ananda Theertha Suresh. 2021. Shuffled model of differential privacy in federated learning. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2521–2529.

[13] Song Han, Huizi Mao, and William J Dally. 2016. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *International Conference on Learning Representations(ICLR)*.

[14] Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems(NeurIPS)*, Vol. 28.

[15] Alexander Herzog, Robbie Southam, Ioannis Mavromatis, and Aftab Khan. 2024. FedMap: Iterative Magnitude-Based Pruning for Communication-Efficient Federated Learning. *arXiv preprint arXiv:2406.19050* (2024).

[16] Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparametrization with gumble-softmax. In *International Conference on Learning Representations(ICLR)*. OpenReview. net.

[17] Yuang Jiang, Shiqiang Wang, Victor Valls, Bong Jun Ko, Wei-Han Lee, Kin K Leung, and Leandros Tassiulas. 2022. Model pruning enables efficient federated learning on edge devices. *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[18] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. 2021. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning* 14, 1–2 (2021), 1–210.

[19] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. (2009).

[20] Yann LeCun, John Denker, and Sara Solla. 1989. Optimal brain damage. In *Advances in Neural Information Processing Systems(NeurIPS)*, Vol. 2.

[21] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip HS Torr. 2019. SNIP: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations(ICLR)*.

[22] Ang Li, Jingwei Sun, Binghui Wang, Lin Duan, Sicheng Li, Yiran Chen, and Hai Li. 2020. Lotteryfl: Personalized and communication-efficient federated learning with lottery ticket hypothesis on non-iid datasets. arXiv:2008.03371 [cs.LG]

[23] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. 2020. Federated Learning: Challenges, Methods, and Future Directions. *IEEE Signal Processing Magazine* 37, 3 (2020), 50–60. https://doi.org/10.1109/MSP.2020.2975749

[24] Xiaolong Ma, Geng Yuan, Xuan Shen, Tianlong Chen, Xuxi Chen, Xiaohan Chen, Ning Liu, Minghai Qin, Sijia Liu, Zhangyang Wang, et al. 2021. Sanity checks for lottery tickets: Does your winning ticket really win the jackpot? *Advances in Neural Information Processing Systems(NeurIPS)* 34 (2021), 12749–12760.

[25] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. 2017. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations(ICLR)*.

[26] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 1273–1282.

[27] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. 2018. Learning Differentially Private Recurrent Language Models. In *International Conference on Learning Representations(ICRL)*.

[28] Jun-Hyung Park, Yeachan Kim, Junho Kim, Joon-Young Choi, and SangKeun Lee. 2023. Dynamic Structure Pruning for Compressing CNNs. In *The Thirty-Seventh AAAI Conference on Artificial Intelligence*.

[29] Pytorch. [n. d.]. GUMBEL SOFTMAX In Pytorch documentation. https://pytorch.org/docs/stable/generated/torch.nn.functional.gumbel_softmax.html. [Online].

[30] Yifan Shi, Kang Wei, Li Shen, Jun Li, Xueqian Wang, Bo Yuan, and Song Guo. 2024. Efficient federated learning with enhanced privacy via lottery ticket pruning in edge computing. *IEEE Transactions on Mobile Computing* (2024).

[31] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[32] Jinhyun So, Ramy E Ali, Başak Güler, Jiantao Jiao, and A Salman Avestimehr. 2023. Securing secure aggregation: Mitigating multi-round privacy leakage in federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37. 9864–9873.

[33] Jinhyun So, Chaoyang He, Chien-Sheng Yang, Songze Li, Qian Yu, Ramy E Ali, Basak Guler, and Salman Avestimehr. 2022. Lightsecagg: a lightweight and versatile design for secure aggregation in federated learning. *Proceedings of Machine Learning and Systems* 4 (2022), 694–720.

[34] Hidenori Tanaka, Daniel Kunin, Daniel L Yamins, and Surya Ganguli. 2020. Pruning neural networks without any data by iteratively conserving synaptic flow. In *Advances in Neural Information Processing Systems (NeurIPS)*, Vol. 33. 6377–6389.

[35] MTCAJ Thomas and A Thomas Joy. 2006. *Elements of information theory*. Wiley-Interscience.

[36] Aleksei Triastcyn and Boi Faltings. 2019. Federated learning with bayesian differential privacy. In *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 2587–2596.

[37] Georgia Tsaloli, Bei Liang, Carlo Brunetta, Gustavo Banegas, and Aikaterini Mitrokotsa. 2021. sfDEVA: Decentralized, Verifiable Secure Aggregation for Privacy-Preserving Learning. In *Information Security -International Conference, ISC (Lecture Notes in Computer Science)*. Springer. https://doi.org/10.1007/978-3-030-91356-4_16

[38] Chaoqi Wang, Guodong Zhang, and Roger Grosse. 2020. Picking winning tickets before training by preserving gradient flow. In *International Conference on Learning Representations(ICLR)*.

[39] Danye Wu, Miao Pan, Zhiwei Xu, Yujun Zhang, and Zhu Han. 2020. Towards Efficient Secure Aggregation for Model Update in Federated Learning. In *IEEE Global Communications Conference, GLOBECOM*. IEEE. https://doi.org/10.1109/GLOBECOM42002.2020.9347960

[40] Xiyuan Yang, Wenke Huang, and Mang Ye. 2023. Dynamic personalized federated learning with adaptive differential privacy. *Advances in Neural Information Processing Systems* 36 (2023), 72181–72192.

[41] Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M Alvarez, Jan Kautz, and Pavlo Molchanov. 2021. See through gradients: Image batch recovery via gradinversion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 16337–16346.

[42] Kai Yue, Richeng Jin, Chau-Wai Wong, Dror Baron, and Huaiyu Dai. 2023. Gradient Obfuscation Gives a False Sense of Security in Federated Learning. In *Proceedings on USENIX Security*.

[43] Xinwei Zhang, Xiangyi Chen, Mingyi Hong, Zhiwei Steven Wu, and Jinfeng Yi. 2022. Understanding clipping for federated learning: Convergence and client-level differential privacy. In *International Conference on Machine Learning, ICML 2022*.

[44] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. 2020. idlg: Improved deep leakage from gradients. arXiv:2001.02610 [cs.LG]

[45] Ligeng Zhu, Zhijian Liu, and Song Han. 2019. Deep leakage from gradients. *Advances in Neural Information Processing Systems(NeurIPS)* 32 (2019).

[46] Zhanxing Zhu, Jingfeng Wu, Bing Yu, Lei Wu, and Jinwen Ma. 2019. The Anisotropic Noise in Stochastic Gradient Descent: Its Behavior of Escaping from Sharp Minima and Regularization Effects. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 7654–7663. https://proceedings.mlr.press/v97/zhu19e.html

APPENDIX

This appendix extends the main paper, providing supplemental materials, including additional details and results, which could not be included in the main paper, due to lack of space.

A PROOF OF THEOREM

This appendix provides the proof of Theorem 4.1 in Section 4.1.

A.1 Proof

Equation 2 is the definition of mutual information, which is:

$$\begin{aligned} \mathsf{I}(X;Y) &= \int_{\mathcal{X}\times\mathcal{Y}} log\frac{d\mathsf{P}_{X,Y}(x,y)}{d\mathsf{P}_X(x)\otimes\mathsf{P}_Y(y)} d\mathsf{P}_{X,Y}(x,y) \\ &= \mathsf{H}(X) - \mathsf{H}(X|Y) \\ &= \mathsf{H}(X) + \mathsf{H}(Y) - \mathsf{H}(X,Y) \end{aligned}$$

Hence, Equation 3 can be written as:

$$\begin{aligned} & \mathsf{I}\left(\mathcal{D}_i;\ \boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t\right)_{t\in[T]} \\ & \overset{(a)}{\leq} \sum_{t=1}^{\tilde{\mathcal{O}}} \mathsf{I}\left(\mathcal{D}_i; \boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]} \\ & \overset{(b)}{=} \sum_{t=1}^{\tilde{\mathcal{O}}} \mathsf{I}\left(\boldsymbol{w}_i^t; \boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]} \\ & + \sum_{t=1}^{\tilde{\mathcal{O}}} \mathsf{I}\left(\mathcal{D}_i; \boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \boldsymbol{w}_i^t, \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]} \\ & - \sum_{t=1}^{\tilde{\mathcal{O}}} \mathsf{I}\left(\mathcal{D}_i; \boldsymbol{w}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t]} \\ & \overset{(c)}{\leq} \sum_{t=1}^{\tilde{\mathcal{O}}} \mathsf{I}\left(\boldsymbol{w}_i^t; \boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]} \\ & = \sum_{t=1}^{\tilde{\mathcal{O}}} \mathsf{H}\left(\boldsymbol{w}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]} \\ & + \sum_{t=1}^{\tilde{\mathcal{O}}} \mathsf{H}\left(\boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]} \\ & - \sum_{t=1}^{\tilde{\mathcal{O}}} \mathsf{H}\left(\boldsymbol{w}_i^t, \boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]} \\ & \overset{(d)}{=} \sum_{t=1}^{\tilde{\mathcal{O}}} \mathsf{I}\left(\mathrm{x}_i^t; \boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k\in[t-1]} \end{aligned} \tag{19}$$

Where (a) and (b) are from the chain rule; (c) is from data processing inequality $\mathcal{D}_i \rightarrow \boldsymbol{w}_i^t \rightarrow \boldsymbol{w}_i^t \odot \bar{m}_i^t$ that $\mathsf{I}\left(\mathcal{D}_i; \boldsymbol{w}_i^t \odot \bar{m}_i^t \middle| \boldsymbol{w}_i^t, \left\{\boldsymbol{w}_i^k \odot \bar{m}_i^k\right\}_{k\in[t-1]}\right) = 0$; (d) is from zero conditional entropy and invariant of mutual information, where

$$\mathrm{x}_i^t = \frac{\partial \ell_i(\bar{\boldsymbol{w}}_i^{t-1}; \mathcal{D}_i)}{\partial \boldsymbol{w}} = \frac{1}{B} \tilde{\sum_{b\in\mathsf{B}_i^t}} g_i\left(\boldsymbol{w}_i^t, b\right) \tag{20}$$

b denotes the size of the random samples.

Therefore, the privacy leakage for a single round t is Equation 5:

$$\mathsf{I}\left(\mathrm{x}_i^t; \boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \middle| \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right)$$

We first prove the upper bound for Equation 5.

Based on Equation 2, we have

$$\begin{aligned}
&\mathsf{I}\left(\mathrm{x}_i^t; \boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \middle| \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right) \\
&= \mathsf{H}\left(\mathrm{x}_i^t \middle| \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right) + \mathsf{H}\left(\boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \middle| \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right) \\
&\quad - \mathsf{H}\left(\mathrm{x}_i^t, \boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \middle| \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right) \\
&\leq \underbrace{\mathsf{H}\left(\boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \middle| \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right)}_{\mathsf{A}} + \underbrace{\mathsf{H}\left(\mathrm{x}_i^t \middle| \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right)}_{\mathsf{B}}
\end{aligned}$$

For part A, based on the chain rule of entropy, we have

$$\begin{aligned}
0 \leq &\mathsf{H}\left(\boldsymbol{w}_i^t, \bar{\boldsymbol{m}}_i^t, \boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \middle| \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right) \\
&= \mathsf{H}\left(\boldsymbol{w}_i^t, \bar{\boldsymbol{m}}_i^t \middle| \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right) \\
&\quad + \mathsf{H}\left(\boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \middle| \boldsymbol{w}_i^t, \bar{\boldsymbol{m}}_i^t, \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right) \\
&= \mathsf{H}\left(\boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \middle| \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right) \\
&\quad + \mathsf{H}\left(\boldsymbol{w}_i^t, \bar{\boldsymbol{m}}_i^t \middle| \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t]}\right)
\end{aligned} \tag{21}$$

Due to $\mathsf{H}\left(\boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \,\middle|\, \boldsymbol{w}_i^t, \bar{\boldsymbol{m}}_i^t, \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right) = 0$, and $\mathsf{H}\left(\boldsymbol{w}_i^t, \bar{\boldsymbol{m}}_i^t \,\middle|\, \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t]}\right) \geq 0$, we have

$$
\begin{aligned}
&\mathsf{H}\left(\boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \,\middle|\, \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right) \\
&\leq \mathsf{H}\left(\boldsymbol{w}_i^t, \bar{\boldsymbol{m}}_i^t \,\middle|\, \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right) \\
&\overset{(a)}{\leq} \mathsf{H}\left(\boldsymbol{w}_i^t \,\middle|\, \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right) + \mathsf{H}\left(\bar{\boldsymbol{m}}_i^t \,\middle|\, \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right) \\
&\overset{(b)}{=} \mathsf{H}\left(\mathrm{x}_i^t \,\middle|\, \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right) + 1 - \frac{1}{2\ln 2}\sum_{n=1}^{\infty}\frac{(2\bar{p}_i-1)^{2n}}{n(2n-1)} \\
&\leq \underbrace{\mathsf{H}\left(\mathrm{x}_i^t \,\middle|\, \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right)}_{\mathsf{B}} + 1 - \frac{\bar{p}_i - 1}{2\ln 2}
\end{aligned}
$$

(a) is from conditioning reduces entropy; (b) is from zero conditional entropy and the Taylor series of the binary entropy function in a neighbourhood of 0.5 with the base pruning rate $\bar{p}_i$.

For part B,

$$
\mathsf{H}\left(\mathrm{x}_i^t \,\middle|\, \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right) = \mathsf{H}\left(\frac{1}{B}\sum_{b\in\mathsf{B}_i^t} g_i\left(\boldsymbol{w}_i^t, b\right) \,\middle|\, \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right)
$$

Let $X = \sum_{b\in\mathsf{B}_i^t} g_i\left(\boldsymbol{w}_i^t, b\right)$, $Z = \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}$, $Y = \frac{1}{B}X$. $f_{X,Z}(x,z)$ and $f_{Y,Z}(y,z)$ are the corresponding joint probability density functions.

Then part B can be written as

$$
\begin{aligned}
\mathsf{H}(Y|Z) &= -\int_{\mathcal{Y},\mathcal{Z}} f_{Y,Z}(y,z)\log\frac{f_{Y,Z}(y,z)}{f_Z(z)}\,\mathrm{d}y\,\mathrm{d}z \\
&= -\int_{\mathcal{X},\mathcal{Z}} |B|\, f_{X,Z}(By,z)\log|B|\frac{f_{X,Z}(By,z)}{f_Z(z)}\,\mathrm{d}y\,\mathrm{d}z \\
&= -\int_{\mathcal{X},\mathcal{Z}} f_{X,Z}(x,z)\log\frac{f_{X,Z}(x,z)}{f_Z(z)}\,\mathrm{d}x\,\mathrm{d}z + log\frac{1}{|B|} \\
&= \underbrace{\mathsf{H}\left(\frac{1}{B}\sum_{b\in\mathsf{B}_i^t} g_i\left(\boldsymbol{w}_i^t, b\right) \,\middle|\, \left\{\boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right\}_{k\in[t-1]}\right)}_{\mathsf{C}} + \log\frac{1}{B}
\end{aligned}
$$

In recent theoretical results for analyzing the behaviour of SGD, they approximate the SGD vector by a distribution with independent components or by a multivariate Gaussian vector.

We define $\hat{g}_i\left(\boldsymbol{w}_i^t, b\right) \in \mathsf{R}^{d^*}$ is the largest sub-vector of the $g_i\left(\boldsymbol{w}_i^t, b\right)$ with non-singular convariance matrices, where $d^* \leq d$. Based on the ZCA whitening transformation and Assumption 1, we have $\hat{g}_i\left(\boldsymbol{w}_i^t, b\right) = \Sigma_i^{-\frac{1}{2}}\boldsymbol{z}$, where $\boldsymbol{z}$ has zero mean and I_{d^*} convariance matrix.

Then for the part C, we set $\boldsymbol{v} = \sum_{b \in \mathsf{B}_i^t} \boldsymbol{z}$ and $\boldsymbol{u} = \Sigma_i^{-\frac{1}{2}} \boldsymbol{v}$, then we have:

$$
\begin{aligned}
\mathsf{C} &= \mathsf{H}\left(\sum_{b \in \mathsf{B}_i^t} \hat{g}_i\left(\boldsymbol{w}_i^t, b \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k \in [t-1]} \right) \\
&= \mathsf{H}\left(\Sigma_i^{-\frac{1}{2}} \sum_{b \in \mathsf{B}_i^t} \boldsymbol{z} \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k \in [t-1]} \right) \\
&= -\int \mathsf{P}_{\mathcal{U}}(\boldsymbol{u}) \log \mathsf{P}_{\mathcal{U}}(\boldsymbol{u}) \, \mathrm{d}\boldsymbol{u} \\
&\overset{(a)}{=} -\int \frac{\mathsf{P}_{\mathcal{V}}(\boldsymbol{v})}{\det\ \Sigma_i^{-\frac{1}{2}}} \log \left(\frac{\mathsf{P}_{\mathcal{V}}(\boldsymbol{v})}{\det\ \Sigma_i^{-\frac{1}{2}}} \right) \det\ \Sigma_i^{-\frac{1}{2}} \, \mathrm{d}\boldsymbol{v} \\
&= \log\ \det\ \Sigma_i^{-\frac{1}{2}} + \underbrace{\mathsf{H}\left(\sum_{b \in \mathsf{B}_i^t} \boldsymbol{z} \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k \in [t-1]} \right)}_{\mathsf{D}}
\end{aligned}
\tag{22}
$$

(a) is based on the transformation of random vectors. Based on the Maximum Entropy Upper Bound [35], which the continuous distribution X with prescribed variance $\mathcal{V}(\mathrm{X})$ maximizing the entropy is the Gaussian distribution of same variance. Since the entropy of a multivariate Gaussian distribution with mean μ and covariance $K_{i,j}$ is

$$
\mathsf{h}(\mathcal{N}_n(\mu, K)) = \frac{1}{2} \log\left((2\pi e)^n |K|\right)
$$

where $|K|$ denotes the determinant of K. Hence, the maximum entropy upper bound for part D is

$$
\mathsf{H}\left(\sum_{b \in \mathsf{B}_i^t} \boldsymbol{z} \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k \in [t-1]} \right) \leq \frac{1}{2} \log\left((2\pi e)^{d^*} |\mathsf{I}_{d^*}|\right) = \frac{d^*}{2} \log(2\pi e) \tag{23}
$$

Therefore, by combining Equation 22 and Equation 23, the upper bound for part B is

$$
\mathsf{H}\left(\mathrm{x}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k \in [t-1]} \right) \leq \log \frac{1}{B} + \log\ \det\ \Sigma_i^{-\frac{1}{2}} + \frac{d^*}{2} \log(2\pi e) \tag{24}
$$

After summing part A and part B, we derive the upper bound for the privacy leakage for the single round, which is Equation 5:

$$
\mathsf{I}\left(\mathrm{x}_i^t; \boldsymbol{w}_i^t \odot \bar{\boldsymbol{m}}_i^t \,\middle|\, \boldsymbol{w}_i^k \odot \bar{\boldsymbol{m}}_i^k\right)_{k \in [t-1]} \right) \leq 1 - \frac{\bar{p}_i - 1}{2 \ln 2} + 2 \log \frac{1}{B} + 2 \log\ \det\ \Sigma_i^{-\frac{1}{2}} + d^* \log(2\pi e)
$$

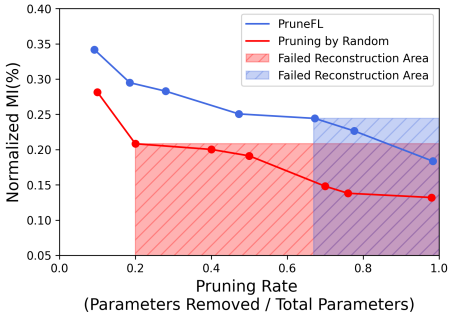


Fig. A1. Impact of Varying Pruning Rates on Reconstructed Image Quality Using Normalized Mutual Information (NMI) Metric in DLG Attack.

Based on Equation 4, we have the upper bound for objective function Equation 3, standing how much information the aggregated model over T global training rounds could leak about the private data,

$$\mathsf{I}_i \leq T - \frac{T(\bar{p}_i - 1)}{2\ln 2} + 2T \log \frac{1}{B} + 2T \log \det \Sigma_i^{-\frac{1}{2}} + d^* T \log (2\pi e)$$

For PruneFL, the upper bound for the single-round leakage is

$$\begin{aligned} \mathsf{I}_i^t &\leq 1 + \frac{\bar{\mathcal{P}} \cup \mathcal{A}}{2d \ln 2} + 2 \log \frac{1}{B} + 2 \log \det \Sigma_i^{-\frac{1}{2}} + d^* \log (2\pi e) \\ &\leq 1 + \frac{\bar{\mathcal{P}}}{2d \ln 2} + 2 \log \frac{1}{B} + 2 \log \det \Sigma_i^{-\frac{1}{2}} + d^* \log (2\pi e) \end{aligned} \tag{25}$$

B PRIVACY ATTACKS

B.1 Extra Implementation details

Our algorithms are implemented by Pytorch and we implement experiments on two NVIDIA RTX A5000 and two Xeon Silver 4316. Across all the examined methods, we employ 1 local training and initialize the learning rate to 0.25 and The baseline pruning rate is set at 0.3 for all datasets.

Data Partitioning. Our data partitioning methodology is aligned with the FEMNIST setting, utilizing its inherent 193-user partition. As for the CIFAR-10 dataset, we divided it into 100 equal-sized, non-overlapping users, following the methodology used in PruneFL [17].

B.2 Comparison of Attacks

In this subsection, we initiate our attack analysis by employing the classic Gradient Inversion (GI) attack [11], a member of the DLG attack family, applied to both the Random and PruneFL pruning methods. As depicted in Figure A1, under this classic DLG attack, the reconstruction of local image data fails even with low pruning rates, such as $p = 0.2$ for Random pruning.

This could be attributed to the sparsity patterns introduced by model pruning in FL. Our exploration takes a step further to optimize the attack strategy. The goal is to capitalize on the characteristics of sparsity inherent to the model pruning context. Consequently, we introduce an advanced attack tailored for the context of model pruning within FL,

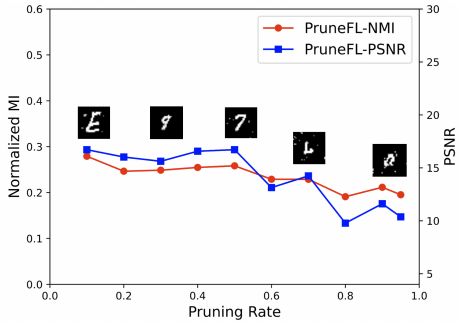


Fig. A2. Impact of varying pruning rate in FEMNIST on privacy leakage using key metrics: Normalized Mutual Information (NMI) displayed on the left y-axis and Peak Signal-to-Noise Ratio (PSNR) shown on the right y-axis under a Gradient Inversion attack for PruneFL.

termed the Sparse Gradient Inversion (SGI) attack. This novel approach aims to recover the pruning mask embedded within the pruned model.

B.3 Evaluation on both NMI and PSNR Metrics

To quantify the extent of privacy leakage, we utilized the Normalized Mutual Information (NMI) metric, which has been demonstrated in prior research to align well with the Peak Signal-to-Noise Ratio (PSNR) [9] as an effective measure of privacy leakage. We examine the relationship between NMI and PSNR. The observed trend, depicted in Fig. A2, illustrates a consistent decrease in both NMI and PSNR metrics with higher pruning rates. This result is consistent with previous research, confirming the reliability of NMI as a measure of privacy leakage, which is comparable to the widely accepted PSNR metric.

C EVALUATION METHODS

C.1 Attack Performance

Considering space constraints, we present the attack performance outcomes for the FEMNIST and CIFAR10 datasets in this section.

FEMNIST Dataset. Figure 4 and Figure 5 show the impact of varying pruning rate, batch size, model size, and training rounds on privacy leakage using Normalized Mutual Information (NMI) under a Sparse Gradient Inversion (SGI) attack for six pruning methods: Random, SNIP, SynFlow, GraSP, FedDST, and PruneFL, on FEMNIST dataset.

Impact of Pruning Rate (p). Figure 4a illustrates the impact of varying pruning rates p from 0.1 to 0.9 on privacy leakage using NMI on the FEMNIST dataset. Consistent with our Theorem 4.1, we observe that higher pruning rates lead to a decreasing trend in NMI. This reduction indicates a lowered risk of privacy leakage due to the fewer parameters present in the pruned model. Figure A3 shows the image reconstruction under different pruning rates. We notice the reconstructions become harder under a large pruning rate.

Impact of Batch Size (B). Figure 4b shows the impact of varying batch size B on privacy leakage using NMI on the FEMNIST dataset, where B is selected from the set 1, 16, 20, 32, 64. It demonstrates that larger batch sizes increase the

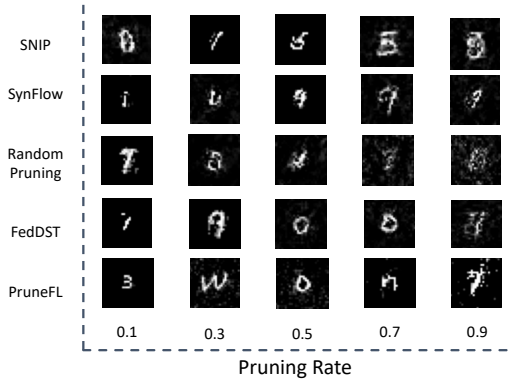


Fig. A3. Impact of varying pruning rate on image reconstruction in FEMNIST using different base pruning methods.

privacy protection of the FL process as they enable more data points to be added during local training, which can help in obscuring the data characteristics of a particular user.

Impact of Model Size (d). We consider 4 architecture of the models: Conv-1, Conv-2, Conv-4, and Conv-6, featuring 1, 2, 4, and 6 convolutional layers, along with 2 linear layers. These models encompass 3,815,566, 6,603,710, 13,657,406, and 25,324,350 parameters, respectively. Figure 5a depicts the relationship between the model size (d) and the corresponding increase in information leakage. The x-axis in the figure represents the number of convolutional layers in each model. This observation can be attributed to the fact that larger models possess a higher number of parameters, thereby amplifying gradient information leakage.

Impact of Communication Round (T). In Figure 5b, the depiction of training rounds T on information leakage reveals a non-linear increase. This observation aligns with Corollary 4.2, which provides an upper bound not strictly indicative of actual performance. The algorithms surpass this bound, allowing for potential non-linear growth with increasing rounds, without contradicting the established theorems.

CIFAR10 Dataset. Figure 4 and Figure 5 show the impact of varying pruning rate, batch size, model size, and training rounds on privacy leakage using Normalized Mutual Information (NMI) under a Sparse Gradient Inversion (SGI) attack for six pruning methods: Random, SNIP, SynFlow, GraSP, FedDST, and PruneFL, on CIFAR10 dataset.

Impact of Pruning Rate (p). Figure 4c illustrates the impact of varying pruning rates on privacy leakage using NMI. By controlling the pruning rate from 0.1,0.3,0.5,0.7,0.9, we examine the trade-off between privacy leakage and the extent of model pruning. Consistent with our Theory 4.1, we observe that higher pruning rates lead to a decreasing trend in NMI. This decrease in NMI indicates a diminished risk of privacy leakage due to the reduced amount of information available in the pruned model.

Impact of Batch Size (B). Figure 4d shows the impact of varying batch size B on privacy leakage using NMI on the CIFAR10 dataset, where B is selected from the set 1, 8, 16, 32. It demonstrates that larger batch sizes increase the privacy protection of the FL process as they enable more data points to be added during local training, which can help in obscuring the data characteristics of a particular user.

Impact of Model Size (d). We consider 4 different model architectures: VGG-4, VGG-6, VGG-9, and VGG-11. These models consist of 1 convolutional layer and 2 linear layers, 4 convolutional layers and 2 linear layers, 6 convolutional layers with 3 linear layers, and 8 convolutional layers with 3 linear layers, respectively. These models encompass 1,125,642, 1,814,538, 6,436,106, and 9,385,994 parameters, respectively. Figure 5c depicts the relationship between the model size (d) and the corresponding increase in information leakage. The x-axis in the figure represents the number of layers in each model. This observation can be attributed to the fact that larger models possess a higher number of parameters, thereby amplifying gradient information leakage.

Impact of Communication Round (T). Figure 5d illustrates the impact of training rounds T on information leakage. Generally, an increase in training rounds results in more information leakage, but it's worth noting that the increase is not linear for all rounds. This observation aligns with Corollary 4.2, which provides an upper bound not strictly indicative of actual performance. The algorithms surpass this bound, allowing for potential non-linear growth with increasing rounds, without contradicting the established theorems.

D DEFENSE DETAILS

D.1 Results of Insights

Within the scope of the three defense methods in the Insights from Evaluation Section, specific configurations have been established. These settings are detailed below:

For all three defense methods, the total pruning rate ($\hat{p}$) is uniformly set at 0.3, mirroring the original PruneFL's pruning rate.

For each defense method:

- PruneFL + Largest: The defense strategy involves pruning based on the weights with top-k largest gradients, with the defense pruning rate set to $\hat{p}_{largest} = 0.3$.
- PruneFL + Random: The defense strategy involves random pruning, with the defense pruning rate ($\hat{p}$) set to $\hat{p}_{random} = 0.3$.
- PruneFL + Mix: This is a hybrid approach that combines both the largest gradient-based pruning and random pruning. The defense pruning rate is determined as the sum of $\hat{p}_{largest} = 0.15$ and $\hat{p}_{random} = 0.15$.

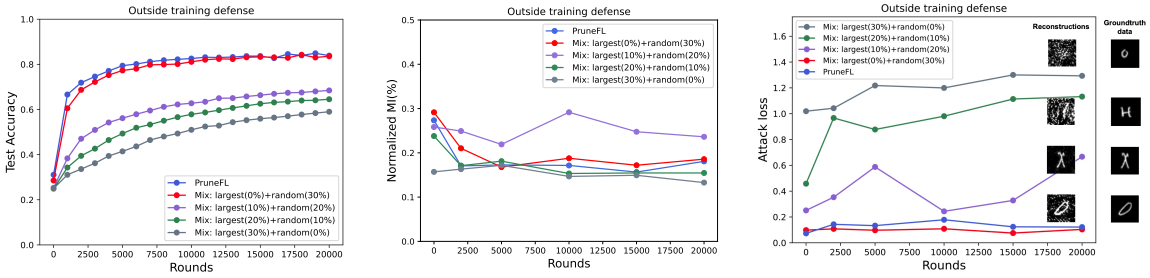
Additionally, we explore various combinations of the pruning rates for largest and random within the Mix strategy, including 0% largest with 30% random, 10% largest with 20% random, 20% largest with 10% random, and 30% largest with 0% random.

Figure A4 shows the performance in terms of accuracy, privacy assessed through NMI, and attack loss of these four combinations across 20,000 iterations of FL. The outcomes reveal that the configuration of 0% largest with 30% random attains the highest accuracy, while 30% largest with 0% random yields the highest level of privacy protection.

Since we focus on privacy in this work, we first proceed to investigate the privacy-accuracy trade-off for the 30% largest with 0% random configurations.

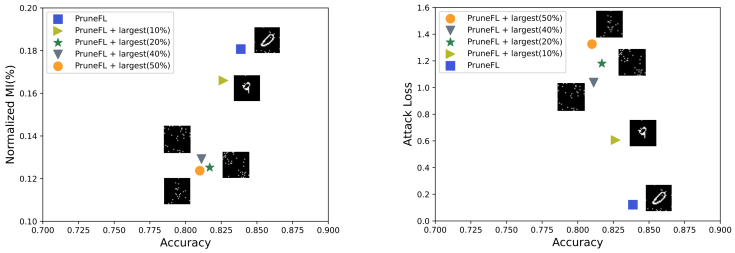
Figure A5 shows Privacy-Accuracy Trade-Off with Varying Defense Rate (0% to 50%) in Largest Method based on PruneFL using NMI and Attack Loss Privacy Metrics.

The outcomes are indicative of a notable trend: as more weights associated with larger gradients are pruned, privacy is enhanced at the expense of accuracy. This substantiates our **Insight 1**: Largest Weights Matter. Pruning weights with large gradients improves privacy the most but also hurts model accuracy the most.



(a) Accuracy Performance (b) NMI Performance (c) Attack Loss Performance

Fig. A4. Comparative Performance Analysis of Four Defense Rate Combinations in MIX Strategy across 20,000 FL Rounds.



(a) ACC vs NMI (b) ACC vs Attack Loss

Fig. A5. Comparison of Privacy-Accuracy Trade-Off with Varying Defense Rate (0% to 50%) in Largest Method based on PruneFL using NMI and Attack Loss Privacy Metrics

D.2 Performance on CIFAR10 Dataset

To show the effectiveness of *PriPrune*, we also implement *PriPrune* on CIFAR10 Dataset with various pruning methods as shown in Figure 9b. By incorporating *PriPrune*, all pruning methods successfully elevate its privacy levels without undermining accuracy.

	λ_{pri} = 1			λ_{pri} = 10			λ_{pri} = 15		
Parameter	λ_{sha} = 2e − 04	λ_{sha} = 2e − 05	λ_{sha} = 2e − 06	λ_{sha} = 2e − 04	λ_{sha} = 2e − 05	λ_{sha} = 2e − 06	λ_{sha} = 2e − 04	λ_{sha} = 2e − 05	λ_{sha} = 2e − 06
λ_{acc} = 1	0.166	0.171	0.156	0.138	0.153	0.154	0.138	0.159	0.151
λ_{acc} = 5	0.154	0.158	0.136	0.125	0.121	0.123	0.135	0.123	0.130
λ_{acc} = 10	0.148	0.142	0.140	0.132	0.126	0.118	0.135	0.132	0.140

Table A1. Comparison of privacy levels (NMI) during hyper-parameter selection: with regards to our hyperparameters in loss functions, we search λ_{acc} in the list of [1, 5, 10], λ_{pri} in the list of [1, 10, 15] and λ_{sha} in the list of [2e-06, 2e-05, 2e-04].

D.3 Hyper-parameter Selection

Table A1 has searched λ_{acc} in the list of [1, 5, 10], λ_{pri} in the list of [1, 10, 15] and λ_{sha} in the list of [2e-06, 2e-05, 2e-04], which showing the comparisons of privacy levels during the hyper-parameter search. As shown in Table A1, when we increase the λ_{pri}, the NMI value becomes lower, indicating better privacy protection. We explore different combinations of hyper-parameters and finally utilize the best set of values that could achieve the best privacy-accuracy tradeoff. For different pruning methods and different datasets, we are adopting different combinations of hyper-parameters

so as to achieve better privacy-utility tradeoff. In FEMNIST, we are using $\lambda_{pri} = 15$, $\lambda_{sha} = 2e-05$ and $\lambda_{acc} = 5$ for PruneFL+*PriPrune*, $\lambda_{pri} = 1$, $\lambda_{sha} = 2e-06$, $\lambda_{acc} = 10$ for Synflow+*PriPrune*, $\lambda_{pri} = 1$, $\lambda_{sha} = 2e-06$, $\lambda_{acc} = 10$ for SNIP+*PriPrune* and $\lambda_{pri} = 8$, $\lambda_{sha} = 2e-05$, $\lambda_{acc} = 10$ for FedDST+*PriPrune*. In CIFAR-10, we are using $\lambda_{pri} = 10$, $\lambda_{sha} = 2e-05$ and $\lambda_{acc} = 5$ for PruneFL+*PriPrune*, $\lambda_{pri} = 1$, $\lambda_{sha} = 2e-06$, $\lambda_{acc} = 5$ for Synflow+*PriPrune*, $\lambda_{pri} = 1$, $\lambda_{sha} = 2e-06$, $\lambda_{acc} = 10$ for SNIP+*PriPrune* and $\lambda_{pri} = 10$, $\lambda_{sha} = 2e-05$, $\lambda_{acc} = 10$ for FedDST+*PriPrune*.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009